
university of groningen
faculty of science and engineering
computing science
sc@RUG 2024 proceedings
21st SC@RUG
2023-2024
Rein Smedinga, Michael Biehl (editors)
rug.nl/research/bernoulli

SC@RUG 2024 proceedings

Rein Smedinga
Michael Biehl
editors

2024
Groningen

ISBN (e-pub): 978-94-034-3092-8
Publisher: University of Groningen Press
Title: 21st SC@RUG 2023-2024 proceedings
Computing Science, University of Groningen
NUR-code: 980

About SC@RUG 2024

Introduction

SC@RUG (or student colloquium in full) is a course that master students in computing science follow in the first year of their master study at the University of Groningen.

SC@RUG was organized as a conference for the 21st time in the academic year 2023-2024. Students wrote a paper, participated in the review process and gave a presentation.

SC@RUG is organized by Rein Smedinga and Michael Biehl, both from the Bernoulli institute. Danië lle Fluks (Faculty of Arts) helped with improving the presentaiton skills of the students.

Organizational matters

SC@RUG 2024 was organized as follows:

Students were expected to work in teams of two. The student teams could choose between different sets of papers, that were made available through the digital learning environment of the university, *Brightspace*. Each set of papers consisted of about three papers about the same subject (within Computing Science). Some sets of papers contained conflicting opinions. Students were instructed to write a survey paper about the given subject including the different approaches discussed in the papers. They should compare the theory in each of the papers in the set and draw their own conclusions, potentially based on additional research of their own.

After submission of the papers, each student was assigned one paper to review using a standard review form. The staff member who had provided the set of papers was also asked to fill in such a form. Thus, each paper was reviewed three times (twice by peer reviewers and once by the expert reviewer).[1] Each review form was made available to the authors through *Brightspace*.

All papers could be rewritten and resubmitted, also taking into account the comments and suggestions from the reviews. After resubmission each reviewer was asked to re-review the same paper and to conclude whether the paper had improved. Re-reviewers could accept or reject a paper. All accepted papers[2] can be found in these proceedings.

In his lecture about communication in science, Rein Smedinga explained how researchers communicate their findings during conferences by delivering a compelling storyline supported with cleverly designed graphics. Lectures on how to write a paper, on scientific integrity and on the review process were given by Michael Biehl.

Daniëlle Fluks gave a lecture about presentation skills and gave tutorials in small groups about presentation techniques and speech skills.

Students were asked to give a short presentation halfway through the period. The aim of this so-called **two-minute madness** was to advertise the full presentation and at the same time offer the speakers the opportunity to practice speaking in front of an audience. Daniëlle Fluks, Michael Biehl, and Rein Smedinga were present during these presentations.

The final online conference was organized by the students themselves (from each author-pair, one was selected to be part of the organization and the other doing the chairing of one of the presentations). Students organized the conference by setting up the final program, find a sponsor for the breaks, etc. They also found two keynote speaker, **Wester Coenraads** from **TNO** about building fast, robust APIs with Rust and **Hidde van der Lende and Marc Holterman** from **Belsimpel** about their organization including Gomibo.

The organizing students also created a website for this year's conference. All announcements can be found on **https://www.studentcolloquium.nl/2024/**

The overall coordination and administration was taken care of by Rein Smedinga, who also served as the main manager of *Brightspace*.

Students were graded on the writing process, the review process and the 2 minute madness presentation, the presentation during the conference and on their contribution in the organization of this conference.

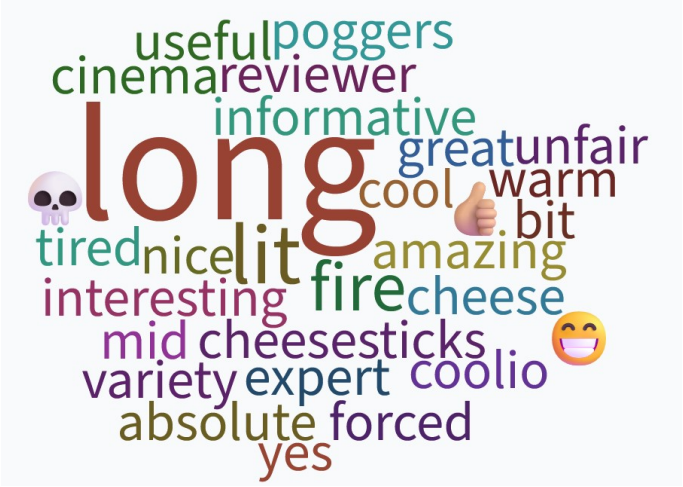
For the grading of the 2 minute madness presentations we used the assessments of the audience using the application *Poll Everywhere* and also used this application to find the best presentation of the day according to the audience.

For the presentations during the conference we also

[1] Since we had a couple of papers written by just one student, some papers were only reviewed by one student and one staff member

[2] This year, all papers were accepted

used *Poll Everywhere* for the assessments of the audiencee (for 50%) and the assessments of Daniëlle Fluks, Michael Biehl and Rein Smedinga (also for 50%). Poll Everywhere again was used to find the best presentation of the day and to ask the audience about their general finding of the symposium, resulting in the following outcome:



The gradings of the draft and final paper were weighted marks of the review of the corresponding staff member (50%) and the two students reviews (25% each).

The complete conference was also recorded and this recording was published on *Brightspace* for self reflection.

The best 2 miniute madness presentation, the best conference proesentation and the best paper were awarded with a voucher and mentioned in the hall of fame.

In contrast to earlier versions, this time we had relatively many (5) individual contributions (i.e., papers written by just one author), and several (3) students who dropped the course halfway.

Website

Since 2013, there is a website for the conference, see **www.studentcolloquium.nl**. The website contains all previous symposium announcements, all available proceedings and the hall of fame (see next page).

Thanks

We could not have achieved the ambitious goals of this course without the invaluable help of the following expert reviewers:

- Sahar Ahmadisakha
- Vasilios Andrikopuolos
- Bochra Boughzala
- Andrea Capiluppi
- Dilek Düstegör
- Daniel Feitosa
- Jiapan Guo
- Christian Kehl
- Jorge A. Perez
- Ayushi Rastogi
- Mirela Riveni
- Huy Truong
- Michael Wilkinson

and all other staff members who provided topics and sets of papers.

Also, the organizers would like to thank Daniëlle Fluks for helping with the presentation skills and the *School of Science and Engineering* for making it possible to publish these proceedings and sponsoring the awards for best presentations and best paper for this conference and our symposium sponsor Belsimpel for providing our keynote presentation and the lunch (cheese stick) during lunch break.

Rein Smedinga
Michael Biehl

Since the tenth SC@RUG in 2013 we added a new element: the awards for best presentation, best paper and best 2 minute madness.

Best 2 minute madness presentation awards

2024
Matej Kucera and Andreea Zelko
Location Inference on Twitter
2023
Germán Calcedo Pérez and Somak Chatterjee
Transferability of Graph Neural Network Generalisation Techniques
and
Shrushti Kaul and Nikhita Prabhakar
Decentralized Federated Learning - Solutions based on Gossip Protocol and Blockchain
2022
David Visscher and Erwin de Haan
A review of networking the cloud datacentre
2021
Niels Bügel and Albert Dijkstra
Mining User Reviews to Determine App Security
2020
Andris Jakubovskis and Hindrik Stegenga
Comparing Reference Architectures for IoT
and
Filipe R. Capela and Antil P. Mathew
An Analysis on Code Smell Detection Tools and Technical Debt
2019
Kareem Al-Saudi and Frank te Nijenhuis
Deep learning for fracture detection in the cervical spine
2018
Marc Babtist and Sebastian Wehkamp
Face Recognition from Low Resolution Images: A Comparative Study
2017
Stephanie Arevalo Arboleda and Ankita Dewan
Unveiling storytelling and visualization of data
2016
Michel Medema and Thomas Hoeksema
Implementing Human-Centered Design in Resource Management Systems
2015
Diederik Greveling and Michael LeKander
Comparing adaptive gradient descent learning rate methods
2014
Arjen Zijlstra and Marc Holterman
Tracking communities in dynamic social networks
2013
Robert Witte and Christiaan Arnoldus
Heterogeneous CPU-GPU task scheduling

Best presentation awards

2024
Mohammad al Shakoush
Sentiment Analysis for Cost Management Actions Related To Related To Changes In IaC Artifacts
2023
Germán Calcedo Pérez and Somak Chatterjee
Transferability of Graph Neural Networks leveraging Graph Structures
2022
Luc Pol and Jeroen Lammers
A High-Level Overview of Minimum Graph-Triangulation Approaches
2021
Niels Bügel and Albert Dijkstra
Mining User Reviews to Determine App Security
2020
none, because of corona virus measures no presentations were given
2019
Sjors Mallon and Niels Meima
Dynamic Updates in Distributed Data Pipelines
2018
Tinco Boekestijn and Roel Visser
A comparison of vision-based biometric analysis methods
2017
Siebert Looije and Jos van de Wolfshaar
Stochastic Gradient Optimization: Adam and Eve
2016
Sebastiaan van Loon and Jelle van Wezel
A Comparison of Two Methods for Accumulating Distance Metrics Used in Distance Based Classifiers
and

Michel Medema and Thomas Hoeksema
Providing Guidelines for Human-Centred Design in Resource Management Systems
2015
Diederik Greveling and Michael LeKander
Comparing adaptive gradient descent learning rate methods
and
Johannes Kruiger and Maarten Terpstra
Hooking up forces to produce aesthetically pleasing graph layouts
2014
Diederik Lemkes and Laurence de Jong
Pyschopathology network analysis
2013
Jelle Nauta and Sander Feringa
Image inpainting

Best paper awards

2024
Andreea-Cristina Zelko and Matej Kucera
Location Inference on Twitter
and
Dogukan Tuna and Kaitlin Vos
Analyzing the Onboarding Experience of Students in Computing Science at the University of Groningen
2023
Xiayo Guan and Lonneke Pules
Machine Learning for Leak Detection in Water Networks
and
Eelke Landsaat and Johanna Lipka
Logistic Regression and Linear Discriminant Analysis: A Comparative Overview and an Empirical Time-Complexity Analysis
and
Mike Lucas and Elnur Seyidov
What makes a great software team?
2022
Erblin Ibrahimi and Sven Hofman
State of the Art: Performance Overview of Black-Box Web Application Scanners
and
Willard Verschoore and Gerrit Sijberen Luimstra
Is it Not Yet Time to Swish? Comparing the ReLU and Swish Activation Functions
2021
Ethan Waterink and Stefan Evanghelides
A Review of Image Vectorisation Techniques
2020
Anil P. Mathew and Filipe A.R. Capela
An Analysis on Code Smell Detection Tools
and
Thijs Havinga and Rishabh Sawhney
An Analysis of Neural Network Pruning in Relation to the Lottery Ticket Hypothesis
2019
Wesley Seubring and Derrick Timmerman
A different approach to the selection of an optimal hyperparameter optimisation method
2018
Erik Bijl and Emilio Oldenziel
A comparison of ensemble methods: AdaBoost and random forests
2017
Michiel Straat and Jorrit Oosterhof
Segmentation of blood vessels in retinal fundus images
2016
Ynte Tijsma and Jeroen Brandsma
A Comparison of Context-Aware Power Management Systems
2015
Jasper de Boer and Mathieu Kalksma
Choosing between optical flow algorithms for UAV position change measurement
2014
Lukas de Boer and Jan Veldthuis
A review of seamless image cloning techniques
2013
Harm de Vries and Herbert Kruitbosch
Verification of SAX assumption: time series values are distributed normally

Contents

Studying the Adoption of Mastodon: A Systematic Literature Review

Eduard Sabo, Tim Gesthuizen

Abstract—The acquisition of Twitter left many users dissatisfied with the change in ownership. As a result, many users transitioned to its competitor Mastodon. This generated interest in researchers, who found themselves in a situation where they could observe and gather data in almost real time. This proved to be an opportunity to study the large migration from an established centralized social media platform to a decentralized network. The existing research investigates various topics and partially finds different results for similar questions. As such, we have conducted a comparative review of the existing findings, extracting well-proven facts and identifying controversial topics. Furthermore, the general empirical discipline and integrity of the papers shall be assessed. While the hands-on research that has been published is rather limited, there is a lack of evaluation of existing research. As such, we aim to address this potential lack of criticism in this study. Moreover, we plan to provide our own contribution to the adoption of new users on Mastodon by studying an already existing dataset. We will perform literature research in the inspected fields and list the general findings of the authors. Already there are topics identified where there are questionable results and the research methodology is disputable. Our goal is to compare the quality of research done regarding decentralized open-source networks, with our primary focus on Mastodon and its adoption by general social media users. Further investigation and own measurements shall identify these kinds of results and assert whether they are defensible. We have found papers with questionable methodology, indicating room for improvement when it comes to the quality of research regarding decentralized networks and Mastodon. Furthermore, a comparison of research and statistical analysis shows that some results are not statistically significant or are at least debatable. In our contribution to this subject, we want to find out the adoption rates of newly joined users during the timeframe of change of ownership at Twitter.

Index Terms—Mastodon, social networks, decentralized systems

1 Introduction

Mastodon is a decentralized online social network (DOSN) that functions as a distributed micro-blogging service. It is meant to offer a free and open-source alternative to services like Twitter. While users can share "Tweets" on Twitter, users of Mastodon can interact through so-called "Toots".
In contrast to established services, Mastodon relies on a decentralized network: There is no central server distributing messages, and users can host their own instances that exchange messages in a mesh-like fashion. This way users of the platform are not relying on the company to provide the service. A good example is the switch of the platform Gab — known for the distribution of hate speech, conspiracy theories and racist ideas — to the Mastodon network. Mastodon issued an official statement that they in no way want to be related to Gab and most Mastodon instances stopped collaborating with the Gab instances. Thus resulting in Gab running in an isolated network.

In order to implement the mentioned exchange of messages between instances, Mastodon implements the ActivityPub protocol. ActivityPub is a platform-independent protocol for synchronization of decentralized social networks [9]. It is standardized by W3C. This way Mastodon even allows for interoperability with other services like personal blogs or other networks.

Instead of connecting to a central server, users connect to so-called instances to post their messages. In order for users of different instances to see each other's posts, instances synchronize messages between each other. This way there is no single point of failure in the network and users are not at the grace of a single instance provider to use the network.

Mastodon started to gain popularity among social media users when Elon Musk acquired Twitter in 2022. Many Twitter users were unhappy with him as the deciding person behind Twitter, given his opinions. As a consequence, many migrated to Mastodon, which did not only offer an alternative, but also did structurally not allow for a similar situation to happen. The case of the network Gab shows a different motivation for a decentralized network. Not only did it allow to keep the platform running with relatively low effort, it also shifted the judicial responsibility for the content of the network away from a single actor.

In our paper, we will make a review of the current existing literature and perform our own analysis on an existing dataset. For these goals, we have the following research questions:

1. RQ1: Does the existing literature differ from the period before and after the acquisition of Twitter?
2. RQ2: What is the adoption rate of users on Mastodon, across 5 instances?

We will summarize the existing State of the Art in section 2. In section 3 we will explain the methodologies that were used to study Mastodon from the related works and we will talk about how we will analyze the dataset that we have. In section 4 we discuss the results and in section 5 we showcase the results from our own analysis. We write conclusions in section 6 and in section 7 we finalize with potential future work topics.

2 State of the Art

In this section, we summarize the research done on the topic of Mastodon. Researchers focused on the study of the decentralized nature of Mastodon, an analysis of its network and the migration of users from centralized platforms to Mastodon.

Lee and Wang conducted the first study on the uses and gratifications obtained from Mastodon. The main concern of users is the privacy of their data, and they argue that Mastodon was created to be a decentralized alternative to mainstream social media platforms with a clear focus on how data privacy should be handled [8].

- *Eduard Sabo is a Software Engineering & Distributed Systems student at University of Groningen, E-mail: e.sabo@student.rug.nl*
- *Tim Gesthuizen is a Software Engineering & Distributed Systems student at University of Groningen, E-mail: t.gesthuizen@student.rug.nl*

Nicholson et al. studied the code of conduct that Mastodon uses. The authors categorized the rules found across the most popular instances on Mastodon and compared them with categories of rules found on Subreddits of Reddit. The researchers would filter out the prescriptive and restrictive rules that could account for multiple categories. The findings show that the rules on Mastodon emphasize the need to create a safe space for its users where harassment and hate speech are limited as much as possible.

Al-khateeb researched the biggest instance of Mastodon - mastodon.social. He looked at the most popular hashtags and the effect of bots on trending hashtags. Moreover, Al-khateeb [4] analyzed the location of users on this instance and performed sentiment and toxicity analysis to understand better the user behaviour of the most popular Mastodon instance. The findings show that DOSNs give users more freedom in what they view on the feed, avoiding featured content takeover, as Mastodon does not use content recommender algorithms that the centralized platforms use.

Nobre et al. performed an image analysis of pictures shared on Mastodon. The researchers retrieved data from the federated timeline of Mastodon, and they studied the category of images that were shared. Moreover, they compared the profiling of images on Mastodon with images from other areas of the Web and found that the majority of pictures posted on Mastodon originate from centralized platforms [11].

2.1 Decentralization of Mastodon

Network decentralization has been an interesting topic that has been researched in various studies. Raman et al. find that while the technology allows for decentralized and redundant networks to be built in Mastodon and similar services, the majority of users rely on central servers in practice [12]. Furthermore, the network of Mastodon instances was not well connected: outages of important Mastodon instances were not only observed but also their impact was found to be quite significant.

Studies on the social interaction between users on Mastodon and how they differ from the interaction examined on centralized social networks have been researched by Zignani et al. and Zulli et al. The former study was the first large-scale analysis of Mastodon at the time and investigated the structure between instances [15], whereas the latter employed 6 different methodologies in their research [17]. Zignani et al. followed up on their study by looking at the repercussions of decentralized architecture, particularly within distributed servers, within the area of DOSNs [16].

2.2 Network Analysis of Mastodon

Many publications perform an analysis of the Mastodon network. One of the most established research about Mastodon is from La Cava, Greco, and Tagarelli [7]. This type of analysis can have different motives. Researchers can try to find out which parts of the network show interest in different types of content. Another use case is to analyze whether the distributed network structure is reflected in other properties of the Mastodon network. Researchers trying to show that the Mastodon network possesses a certain property might back up this claim by showing that it also holds for each instance and not just the whole network.

A common problem that affects us and other researchers are users that are active, but do not perform any actions that can be registered. These users — commonly called "lurkers" – actively use Mastodon but never react to the content they consume. This needs to be kept in mind when collecting data about the network.

2.3 Migration to Mastodon

The newest research topic about Mastodon is the migration of users from Twitter following the acquisition of the popular social media network. For example, Zia et al. [14] and Raman et al. [12] research direct effects of the transition. Our data analysis will also show that the adoption by Musk is measurable in the user data. Finally, Zia et al. [14] discovered that users post different content on Mastodon and Twitter: only 1.53% of users post identical content on both platforms. The content and topics of the different posts were not assessed though. Another example is the study by Jeong et al. [2]. They queried users who have switched for their reason to do so. Furthermore, they found that Twitter is used for debating social issues, while Mastodon is used primarily to discuss niche topics. These findings and the media attention given to the events motivate a rise in research on the topic.

3 Methodology

In our paper, our primary objective is to conduct a comprehensive review of the methodologies employed in the papers identified during the literature review, specifically focusing on the steps involved in the data-gathering process, including data collection, data cleaning, and data processing. We categorize these papers into two major groups: those written and published before the takeover of Twitter, a period when Mastodon was less recognized; and those written and published after the change in ownership at Twitter, a time when research increasingly centred around this pivotal event, capturing heightened scientific interest.

Our investigation aims to examine the existing literature and compare the research conducted on DOSNs, with a specific emphasis on Mastodon, before and after Twitter's takeover. We seek to determine whether the research on Mastodon has exhibited any changes in quality following this transition. To achieve this, we will analyze the papers, scrutinizing their methodologies and findings.

3.1 Methods used in papers before the takeover of Twitter

Below, we describe the methodology of the papers that were published before the takeover of Twitter. All papers published before October 2022 will be part of this category.

Zulli, Liu, and Gehl [17] 6 methods of data collection: 1) interviews with 5 administrators of mastodon instances 2) observations on 3 mastodon instances 3) mastodon toots related to specific hashtags 4) codes of conduct of 400 mastodon instances 5) blog posts and forum discussions of developers of Mastodon 6) technological news articles about Mastodon.

La Cava, Greco, and Tagarelli [7] The paper uses a public dataset published in 2018 in order to perform its analysis. Furthermore, they implemented a web crawler, fetching information recursively through Mastodon's public REST API. The initial seed for the crawler was generated through `instances.social`, a compendium of active and public Mastodon instances. This way, the authors were able to collect data about 81,000 new users.

La Cava, Greco, and Tagarelli [6] The authors measure degree assortativity, a metric outlined in two other papers, giving insight into user relationships. They collected the data using Mastodon's public API and verified them through transitive properties. The authors reduced the studied data set by showing that their subset has similar properties as the whole network and limited themselves to these 5 instances, removing too "noisy" ones. Finally, the authors compared the different instances using statistics from `instances.social` and `fediverse.party` and discussed their findings with regards to differences between Mastodon instances, how decentralized

Mastodon as a service is and what makes the special unique from a users perspective.

Zignani, Gaito, and Rossi [15] The authors decided to perform their analysis using data queried from the social networks' public API after evaluating different approaches to obtain data. Just like, La Cava, Greco, and Tagarelli [7], they used a custom web crawler in addition to obtain further data for their study. Afterwards, the authors establish the "following" relationship: Users following another user forms a large directed graph out of the network. Graph theory can be applied to identify components in the graph and deduce information about the structure of the network, also with regard to the network's instances. This dataset is the main contribution of the paper.

Zignani et al. [16] The paper uses their previously used dataset Zignani, Gaito, and Rossi [15] to perform their studies. In contrast, it does some more extensive analysis of the data though. In particular, the authors investigate how big of an impact instances have on a user's experience of the Mastodon network. Furthermore, they use self-designed formulas to quantify the impact of single instances on the whole network. Just like some prior publications, the research aims to show that the theoretical abilities of Mastodon, to provide privacy and a decentralized, failure-robust social network, are less effective in practice regarding the actual social network.

Al-khateeb [4] The author analyzes trending hashtags on Mastodon with regards to the users involved, their location and what they share in general. For this purpose, the author collected data through Mastodon's public API for 35 days using publicly available Python scripts. Afterwards, the author analyzed the hashtags that were trending in the inspected time frame. Next, the posts are classified using AI technology. In particular, Google Perspective API is used to calculate the toxicity of the respective posts. As a consequence, the service dictates the metric of what constitutes a toxic post. This score is used to compare the toxicity of posts. However, this is not a feature of the service: The service calculates how certain it is that a post is toxic, not how toxic it is. In addition, at the time of writing the model is only advertised to detect "severe toxicity, insults and profanity"[1].

Nobre, Ferreira, and Almeida [11] The authors collect data from Mastodon's public API and filter it for explicit images and try to answer how this data is shared between instances. One problem was the fact that most images are linked to on Mastodon, but are actually stored by external services. The authors use Google's Cloud Vision API to judge whether an image contains adult content. Next, they try to identify the source of the image through Google's Image Search. Using this data the authors can determine whether content was first shared on Mastodon or whether it was propagated from somewhere else and how it spread in the social network. Furthermore, they try to identify when content was first shared on other social media websites before it was posted on Mastodon. They use this data for their discussion and result analysis.

Raman et al. [12] The authors focus on analyzing the decentralized nature of Mastodon. For their study, the authors gathered data through Mastodon's API about a list of instances obtained from `mnm.social`. The goal here was to analyze who hosts their own instances and how decentralized the Mastodon landscape is. Furthermore, the authors analyzed the impact of certain Mastodon instances failing. Lastly, they cross-referenced their processed data with existing Twitter data from `pingdom.com` and Internet Archive [1].

[1] According to `https://perspectiveapi.com/how-it-works/`

3.2 Methods used in papers after the takeover of Twitter

In this section, we approach the methodologies identified in the papers that were written and published after the takeover of Twitter. All papers published since January 2023 will be part of this category.

Lee and Wang [8] The research employed a Questionnaire Survey approach to conduct a limited-scale exploratory inquiry into the motivations of Mastodon users. The online survey, comprising nineteen questions, applied the Uses and Gratifications perspective. Initially, 254 participants were enlisted from 15 Mastodon instances, with the final count being 150. Data collection took place between February 2021 and March 2021. The questionnaire covered demographic details, sought gratifications (47 items), and obtained gratifications (47 items), using a 5-point Likert scale for responses. The investigation delved into reasons for the adoption of Mastodon, with a particular focus on aspects like privacy and free speech concerns.

Jeong et al. [2] The authors collected data from instances of Mastodon and Twitter using the public API of both platforms. They cross-referenced Mastodon accounts with Twitter accounts. This resulted in 0.6 million toots and 0.4 million boosts from Mastodon, and 1.1 million tweets and 0.5 million retweets from Twitter. After the data cleaning process, the number reached 10.000 users, that were found to have migrated from Twitter to Mastodon.

Kennedy et al. [3] The researchers looked at posts on Instagram and Twitter based on the #dermatology hashtag. They would retrieve the location of the accounts that created a post with this hashtag and would study users who are from the US, UK, Canada and the rest of the world.

Nicholson, Brian, and Fiesler [10] In March 2023, the study compiled a dataset by Web scrapping the top 1,000 Mastodon instances based on active users from the `instances.social` website. The initial dataset included 495 unique instances and 3,503 individual rules. Data collection involved retrieving rules from the "Server Rules" section of each instance's publicly accessible "About" page. Exclusions were made for instances that were down, had non-English rules, or lacked listed rules, resulting in the final dataset. Using a codebook from a prior study, the first author iteratively examined rules from the top 50 instances, incorporating new categories that appeared frequently. The coding process continued until saturation, and the first author manually coded the 100-instance sample for consistency, focusing on understanding rule types, their relationships within instances, and connections to rules found in related research.

La Cava, Aiello, and Tagarelli [5] The research centred on the #TwitterMigration movement, conducting extensive data gathering from both Twitter and Mastodon. Using Twitter's API, tweets related to the migration were collected between October 26th, 2022, and January 19th, 2023. Relevant migration-related hashtags were identified through trending hashtags and snowball sampling, resulting in a final set of 13 chosen hashtags. To connect Twitter users with their corresponding Mastodon profiles, information promoting Mastodon handles on Twitter was employed. Regular expressions were formulated to detect potential Mastodon-like handles in usernames, descriptions, or tweets. A total of 108k handles were identified, and refined to approximately 75k after matching them with known Mastodon instances. For each Mastodon handle, the official Mastodon API was queried to gather follower and followee lists, along with profile metadata. This process aimed to reconstruct users' Twitter social networks on Mastodon.

Zia et al. [14] The paper collects data on the migration of Twitter users to Mastodon after the change of ownership at Twitter in October 2022. First, the authors collected data through Mastodon's

Table 1: Summary of data collected from all five instances

Mastodon instance	Amount of data collected
mastodon.social	87,210 accounts (8% of userbase)
mastodon.cloud	125,329 accounts (50% of userbase)
mstdn.social	30,728 accounts (15% of userbase)
mastodon.online	23,885 accounts (13% of userbase)
mastodon.world	21,608 accounts (13% of userbase)

public API. A list of Mastodon instances is compiled, and tweets containing links to these instances, along with specific migration-related keywords and hashtags, are gathered using Twitter's Search API. The study identifies 136,009 Mastodon accounts of migrating Twitter users, created across 2,879 instances. Tracking Mastodon activity reveals a substantial increase in registrations, logins, and statuses after Musk's Twitter acquisition. Twitter and Mastodon timelines of migrating users are crawled, resulting in a total of 16,163,600 tweets and 5,746,052 Mastodon statuses. Followee data for both platforms is also collected, covering 13,068 users and 11,453,484 followee relationships. The methodology sheds light on migration patterns, user activities, and relationships in the context of the Twitter-Mastodon transition.

3.3 Data Collection

We perform analysis on a dataset from the research of Sabo, Riveni, and Karastoyanova [13]. They have collected a total of 288.760 user account data across 5 instances. In Table 1 we show the amount of user data collected across the 5 instances.

Each user data contains multiple fields, such as ID, username, followers, following, number of posts, date of account creation and date of the last post. In this paper, we will look at the last 2 data fields. We will study the time from the account creation to the time of the last post. The dataset was collected between late May - early June 2023, therefore those are the latest dates collected from which we extracted the time of the last post of a user. Our goal is to find the adoption rate of the users on their respective Mastodon instances so that we have an overview of how many users are actively using Mastodon.

4 Results and Discussion on Literature Review

DOSNs have existed since 2008, when *identi.ca* was first established on the Fediverse. DOSNs were created with the intent to be a niche social media platform for people who wanted to have more appropriate control of their data privacy when they spent time on social networks. However, since the takeover of Twitter, there has been an abundant number of people who migrated from centralized platforms, and with the rise in the user population of Mastodon, the largest DOSN within the fediverse, it also came with a rise in the number of research papers.

We discussed several papers looking at the structure of the Mastodon social network and how it compares to Twitter. Many of the discussed papers come to the conclusion that there are significant differences between Mastodon and Twitter [11] [17] [2]. Furthermore, many papers conclude that while Mastodon can be run in a stable, decentralized network, the majority of users rely on central instances [14]. As a consequence, the practical Mastodon network does not possess all the distributed properties it could have. For example, the network is much less fault tolerant than it could be, and instance outages, that do happen in practice, do affect the stability of the overall network [12]. Finally, we notice some differences in the scientific work before and after the acquisition of Twitter, as outlined below.

4.1 Results from papers before the takeover of Twitter

In the Table 2, we show the results of the literature summarized in the state of the art which belongs to the group of authors who studied Mastodon before the acquisition of Twitter.

Researcher	**Results**
Zulli, Liu, and Gehl [17]	Users adopt Mastodon due to their specific interest in one or more particular instances.
La Cava, Greco, and Tagarelli [7]	Instances have not changed the location of their networks. The federative mechanism of Mastodon makes it stand out compared to other centralized social networks.
La Cava, Greco, and Tagarelli [6]	Classification of Mastodon structure by assigning users as lurkers and bridge. Users adopt Mastodon due to their specific interest in one or more particular instances
Zignani, Gaito, and Rossi [15]	Users adopt Mastodon due to their specific interest in one or more particular instances.
Zignani et al. [16]	There is a bigger impact on unique effects on degree distribution, clustered structures and user connections in DOSNs compared to their centralized counterparts.
Al-khateeb [4]	On Mastodon users are less restricted than centralized platforms in the type of content they post.
Raman et al. [12]	50% of posts on some Mastodon instances can be removed if those instances are affected by outages.
Nobre, Ferreira, and Almeida [11]	60% of collected pictures belong to one of 5 categories: racy, adult, spoof, medical and violence, and their place of origin is from centralized platforms

Table 2: Data Collection by Researchers Before the Acquisition of Twitter

4.2 Results from papers after the takeover of Twitter

In the Table 3, we show the results of the literature summarized in the state of the art, which corresponds to the researchers that studied Mastodon after the change of ownership at Twitter.

4.3 Discussion

To answer RQ1, we find significant differences in how the authors studied Mastodon in these 2 distinct periods. The literature before the Twitter acquisition mainly studies Mastodon as a social network and its decentralized nature. Furthermore, it makes comparisons between Twitter and Mastodon and would treat this as a comparative analysis between a DOSN and a centralized platform, with a focus on the differences between the two choices and the unique features that a federated network offers. Here we identify 2 groups of researchers who established themselves in this topic: Zignani et al. and La Cava et al.

Regarding the studies conducted after the aquistion of Twitter, La Cava et al. continues their work from the previous 2 papers, and Lee and Wang dive into a uses and gratifications analysis of Mastodon. The

Researcher	Result
Lee and Wang [8]	Main drivers of users adopting Mastodon are ”Convenience”, ”Privacy” and ”Social Escapism and Social Support”
Jeong et al. [2]	User behaviour and Mastodon’s unique features contribute to user adoption
Kennedy et al. [3]	Regarding the online dermatology community, Mastodon can be used as their main social media platform
Nicholson, Brian, and Fiesler [10]	Mastodon instances are oriented around online security and freedom of speech
La Cava, Aiello, and Tagarelli [5]	The number of social connections is lower when the migration process brings in larger numbers of users.
Zia et al. [14]	2.26 % of Mastodon users who migrated deleted their Twitter accounts. Mastodon is less toxic than Twitter.

Table 3: Results by Researchers After the Acquisition of Twitter

other authors focus on the migration of users between Twitter and Mastodon.

5 Data Analysis Results

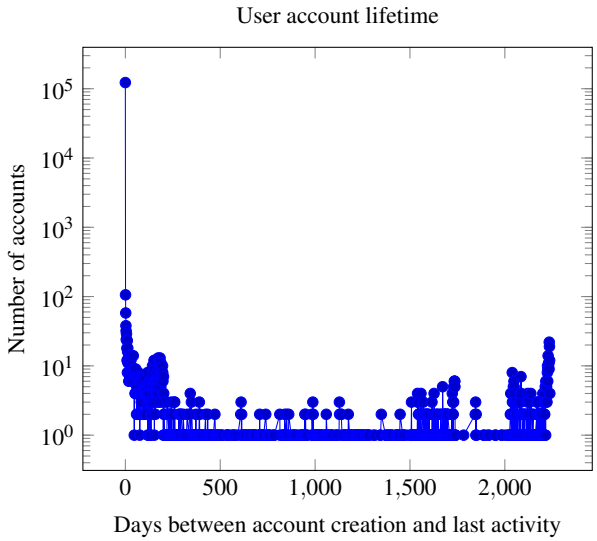


Fig. 1: Data for `mastodon.cloud`

To answer RQ2, we used the existing dataset and analyzed how long users were active on Mastodon. The idea is, to possibly falsify a migration from Twitter to Mastodon given a lack of user activity. The results are visualized in figure 2, 3, 4 and 5. It is noteworthy that a majority of users never post, leading to their y-value being 0. In addition, local peaks at around 200 and 400 can be seen across all instances. While we have no references or any theories regarding this, the 400-day peak could be explained by a larger amount of users creating Mastodon accounts when Twitter was acquired and was active until the beginning of this study. This just happens to be ≈ 420 days at the time of writing.

Lastly, there are some important weaknesses in our analysis. Interestingly, some user accounts have been created way before Mastodon was published. The problem was analyzed and it was concluded that these accounts were imported from the Fediverse into Mastodon. This also explains our longest active account with a span of 5896 days, which coincides nicely with the creation of the Fediverse in 2008.

The data shows a lower bound of the activity spans of users. Users

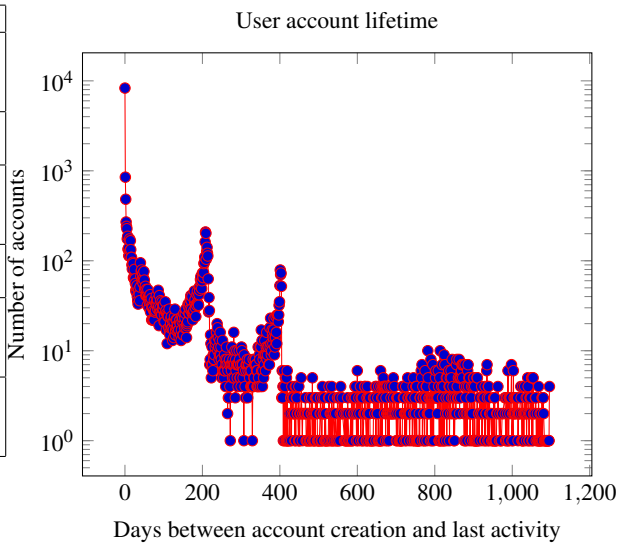


Fig. 2: Data for `mastodon.online`

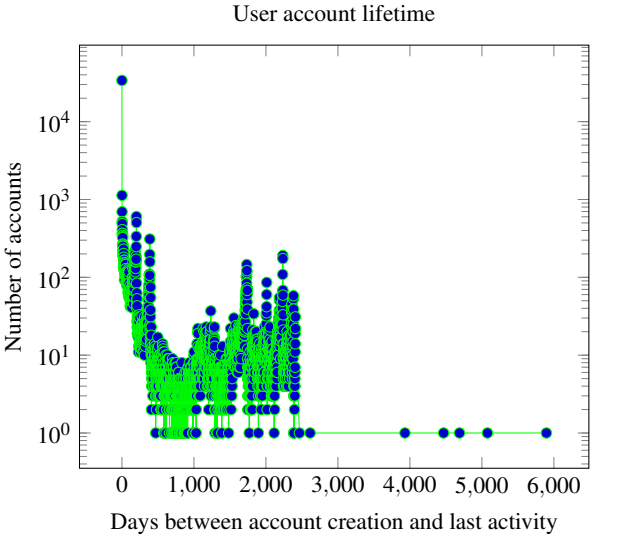


Fig. 3: Data for `mastodon.social`

can be active in the social network without posting themselves, which is not respected in our measurement approach. So-called “lurkers” — users that only consume others’ content without publishing content themselves — are portrayed as inactive users. This also explains the large volume of users with an activity span of 0 days across all instances.

6 Conclusion

We have investigated numerous publications by different authors, both before the acquisition of Twitter and after. In both categories, we have discovered papers with debatable results and methodologies. Furthermore, we have found an increase in scientific publications about Mastodon after the Twitters acquisition. As a consequence, there are observable differences in research, but we cannot prove qualitative changes. Nonetheless, our data indicates that the user migration from Mastodon to Twitter is a real phenomenon and that users who migrated in this time frame are still active on Mastodon.

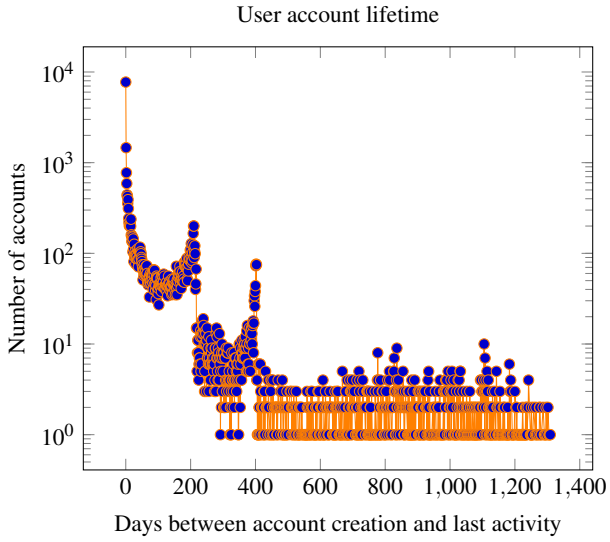


Fig. 4: Data for `mstdn.social`

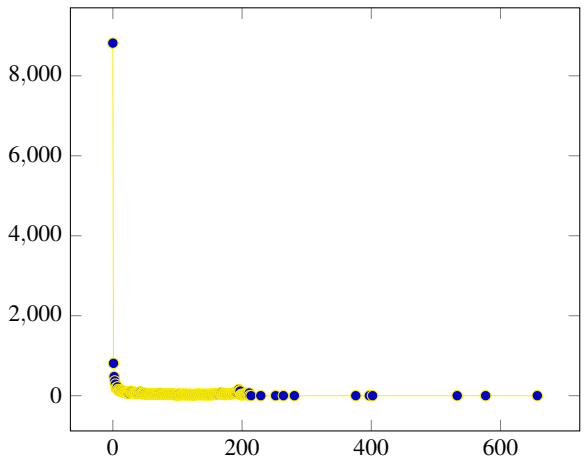


Fig. 5: Data for `mastodon.world`

7 Future Work

We would like to gather more data in the future to expand on the existing research questions and the aspects we could not answer in this paper. Additionally, future research could explore long-term migration trends between centralized platforms like Twitter and decentralized platforms like Mastodon. This could involve examining the movement of specific user groups from more centralized to decentralized social media and the impact of social and political factors on the migration.

References

[1] Internet Archive. *Twitter Outages*. https://web.archive.org/web/20110828003545/http://stats.pingdom.com/wx4vra365911/23773/2007/02. [Accessed on March 24, 2024]. 2007.

[2] U. Jeong et al. "Exploring Platform Migration Patterns between Twitter and Mastodon: A User Behavior Study". In: (2023). arXiv: 2305.09196 [cs.SI].

[3] J. Kennedy et al. "We hardly knew ye... goodbye, #dermtwitter?" In: *Dermatology Online Journal* 29.5 (2023). DOI: 10.5070/D329562422.

[4] S. Al-khateeb. "Dapping into the Fediverse: Analyzing What's Trending on Mastodon Social". In: *Social, Cultural, and Behavioral Modeling*. Cham: Springer International Publishing, 2022, pp. 101–110. ISBN: 978-3-031-17114-7.

[5] L. La Cava, L. M. Aiello, and A. Tagarelli. *Get Out of the Nest! Drivers of Social Influence in the #TwitterMigration to Mastodon*. 2023. arXiv: 2305.19056 [cs.CY].

[6] L. La Cava, S. Greco, and A. Tagarelli. "Information consumption and boundary spanning in Decentralized Online Social Networks: The case of Mastodon users". In: *Online Social Networks and Media* 30 (2022). DOI: 10.1016/j.osnem.2022.100220.

[7] L. La Cava, S. Greco, and A. Tagarelli. "Understanding the growth of the Fediverse through the lens of Mastodon". In: *Applied Network Science* 6.1 (2021). DOI: 10.1007/s41109-021-00392-5.

[8] K. Lee and M. Wang. "Uses and Gratifications of Alternative Social Media: Why do people use Mastodon?" In: (2023). arXiv: 2303.01285 [cs.HC].

[9] C. Lemmer-Webber et al. *ActivityPub*. W3C Recommendation 20180123. World Wide Web Consortium, Jan. 2018. URL: https://www.w3.org/TR/activitypub/.

[10] M. N. Nicholson, K. C. Brian, and C. Fiesler. "Mastodon Rules: Characterizing Formal Rules on Popular Mastodon Instances". In: 2023, pp. 86–90. DOI: 10.1145/3584931.3606970.

[11] Gabriel P. Nobre, Carlos H. G. Ferreira, and Jussara M. Almeida. "More of the Same? A Study of Images Shared on Mastodon's Federated Timeline". In: *Social Informatics*. Cham: Springer International Publishing, 2022, pp. 181–195. ISBN: 978-3-031-19097-1.

[12] Aravindh Raman et al. "Challenges in the Decentralised Web: The Mastodon Case". In: *CoRR* abs/1909.05801 (2019). arXiv: 1909.05801. URL: http://arxiv.org/abs/1909.05801.

[13] Eduard Sabo, Mirela Riveni, and Dimka Karastoyanova. "Decentralized Networks Growth Analysis: Instance Dynamics on Mastodon". In: *Complex Networks & Their Applications XII*. Springer Nature Switzerland, 2024, pp. 366–377. ISBN: 978-3-031-53503-1.

[14] Haris Bin Zia et al. *Flocking to Mastodon: Tracking the Great Twitter Migration*. 2023. arXiv: 2302.14294 [cs.SI].

[15] M. Zignani, S. Gaito, and G. P. Rossi. "Follow the "mastodon": Structure and evolution of a decentralized online social network". In: 2018, pp. 541–550.

[16] Matteo Zignani et al. "The Footprints of a 'Mastodon': How a Decentralized Architecture Influences Online Social Relationships". In: 2019, pp. 472–477. DOI: 10.1109/INFCOMW.2019.8845221.

[17] D. Zulli, M. Liu, and R. Gehl. "Rethinking the "social" in "social media": Insights into topology, abstraction, and scale on the Mastodon social network". In: *NEW MEDIA & SOCIETY* 22.7, SI (July 2020), pp. 1188–1205. ISSN: 1461-4448. DOI: 10.1177/1461444820912533.

Location Inference on Twitter

Andreea-Cristina Zelko Matej Kucera

Abstract— There are both research and commercial uses for personal location data on the Internet, but such data is rarely available. Research attempting to investigate phenomena which require geo-tagged data often falls short due to this problem. However, location data can be obtained using inference from other available data sources. For social media posts, text contents and other contextual information can be used to infer accurate location information.
Our paper will dive into the details of some proposed location inference techniques in order to understand and report their possible uses. We list some input data types which these algorithms use and the results they are capable of achieving. It is important for users of social media to be aware of this so they can effectively control the amount of information they make publicly available. Awareness of this topic is important in order to appreciate the privacy concerns brought forth by the existence of this field.
We explore the viability of applying Large Language Models (LLMs) to text-based location inference in an empirical study. Our results show that LLMs are able to extract information from text with little to no preprocessing. Hence, we point out the general availability of such tools and their potential power when combined with other pre-existing methods. We mention some limitations of the algorithms presented in this paper as well as our LLM-based approach.

Index Terms—Location inference, privacy, social media, spatial data, data mining, geolocation.

1 Introduction

Ever since the advent of the Internet, people have used it to share glimpses of their lives with others. In the beginning, this took the form of written blogs. Later, these advanced into the video format, resulting in the so-called vlogs. Nowadays, social media has taken the internet by storm, providing its users with platforms that are quick and easy to use. Most people use these platforms for socialization, as the name suggests, and for entertainment. Users spend many hours of their day scrolling through their social media feeds. For example, the average Twitter user spends more than 30 minutes a day on the social media giant according to data published by Twitter [2].

It has also been found that most users are unaware of how much personal data is being collected about them while they browse the Internet [5]. This data consists not only of the data directly available from the user's online presence, but also data which can be inferred. Internet users can often be caught unaware by the power of such inference techniques. Inference aims, by definition, to find data which the users did not willingly share. Companies wishing to use such data may use aggregation techniques to create a highly detailed portrait of their users.

However, publicly available platforms such as the aforementioned social media platform Twitter allow for almost anyone to view users' profiles, their posts and other public information. This has lead to an increase in the number of cyberbullying attacks, as well as cases of online stalking. In this ever expanding domain, scientists and researchers try to find a way to put all the information shared by users to good use. To this end, they have explored many ways of localizing events that are getting much attention on Twitter. It is possible to infer the location of an event by finding tweets connected to it. The tweet locations can then be inferred using the location of the users, the GPS location in the tweet metadata, or the actual places mentioned in the tweet itself.

In this paper we will carry out an overview exploration of the current state of the art technologies used to infer the locations of events. Additionally, we will also conduct a short exploratory study to test the feasibility of using Large Language Models (LLMs) for this task. We use a dataset which provides a list of tweets per user and a city-level home location for each user. Finding a dataset proved difficult as Twitter has recently restricted its API usage and our exploration showed that there aren't many twitter datasets available online. With the limitation of requiring geo-tagged data, we resorted to using a dataset from 2010 [6] since it can still serve as a proof of concept even thought it is not recent.

Current techniques use various forms of input to infer location information. There are several methods predicting users' home location on a less granular level, usually inferring city-level information [19, 11]. Methods predicting tweet location usually find more granular information down to geographic coordinates, while also having access to time-based distributions and therefore being able to infer additional data about the user [10]. These are only some examples of methods which are currently available. We will present a more detailed overview of some selected methods in Section 5.

1.1 Research questions

For the purposes of this overview paper, we formulated the following research questions:

- **RQ1** What are the implications of accurate location inference on social media?
- **RQ2** What are the current location inference techniques and which parameters do they use for inference?
- **RQ3** Can Large Language Models be applied to location inference problems?

The structure of this overview paper is as follows. We first cover the design of our study in Section 2. Then, we will introduce the necessary background information in Section 3. Next we will discuss the implications of location inference on social media in Section 4 and give an overview of some location inference approaches in Section 5. In Section 6 we will present our exploratory study to answer RQ3, and finally, in Section 7 and Section 8 we will present our discussion and conclude the paper. Lastly, we elaborate on possible future work in this field of research in Section 9.

1.2 Abbreviations

We list the abbreviations used in this paper below.

- **LLM** - Large Language Model
- **POI** - Point of Interest
- **IR** - Information Retrieval
- **ILF** - Inverse Location Frequency
- **GAM** - Generalized Additive Model

2 Study design

In this study we aim to provide a clear overview of the current state of the art on the topic of location inference from social media, specif-

- *Matej Kucera is a MSc Computing Science student at the University of Groningen, E-mail: m.kucera@student.rug.nl.*
- *Andreea-Cristina Zelko is a MSc Computing Science student at the University of Groningen, E-mail: a.c.zelko@student.rug.nl.*

ically focusing on Twitter. To find existing research in this area, we began our search by finding multiple overview studies. From those, we followed their references and found many papers showcasing existing approaches. We selected ones which showcase different solutions, so that we may compare them and show their strengths. We also searched for recent studies which are not included in these overview papers and mention these to showcase recent advances in the field. We supplement this overview by exploring the implications of location inference on privacy. We find and highlight some studies which show the decrease in privacy caused by the availability of inference methods.

We conduct an empirical study on using Large Language Models for location inference purposes. The goal is to demonstrate their resolving power and the fact that they can extend and empower existing techniques with further capability. We wish to raise awareness about the general availability of these powerful tools and to show an example of how they can be used by to achieve noble or immoral intentions.

3 Background

We will now provide some necessary background information. Covering this is important, as it introduces concepts which will be referenced later in this paper.

3.1 X, formerly Twitter

This paper focuses specifically on analyzing activity on the social media platform X, formerly known as Twitter. For clarity and disambiguation purposes, we refer to it as Twitter throughout this paper. Most papers we reference throughout this paper were written before the name was changed and hence also refer to the platform as Twitter. This is why we chose to do so as well.

3.2 Twitter usage

Twitter has an active community of users. It self-reportedly hosts half a billion active users every month [2]. It is overshadowed by other social networks in user count, but its unique nature and large user base make it an interesting topic for research nonetheless.

Twitter is unique from other social media due to the nature of the content it supports. Posts created by non-paying users, also called tweets, are limited to 280 characters. Premium users have to adhere to a limit of 25000 characters. Since the majority of users fall into the first category, this introduces an entirely unique way of communication. It is common to see abbreviations, slang and other ways of shortening one's sentences in order to fit more information within one tweet. The communication is also largely lacking in formality, hence misspellings and informal language are common. This form of expression poses new challenges for analysis tools which rely on natural language processing, as they often are unable to handle misspelled words and other features found in such informal language.

The amount of information that Twitter carries and creates every day can definitely be a important source for many future projects. Augmenting this dataset with additional information can provide a valuable tool for further investigations which may otherwise not be possible. There are multiple kinds of location information which can be added, all of which can lead to higher usability of this data for research purposes.

3.3 Location types

There are many approaches which have been proposed for location inference using Twitter's user-generated content. In order to gain an understanding of the field, one must first understand the differences among location types being inferred. On Twitter, one can associate three location types with a user or a tweet. These are:

User location - This is the home location of the user. Twitter allows users to optionally specify their home location in their profile. There is no enforcement over the contents of this field, so users may enter false information for humorous or other reasons. They may enter any combination of characters, so this field can oftentimes contain data completely unrelated to location. The true user location can be incredibly valuable as it provides personal information about the user's home address and can therefore be used in a variety of ways, as mentioned in Section 4. This location can be inferred in various ways, as will be discussed later.

Tweet location - This represents the location from which a tweet has been posted. This information is also optionally included with each tweet, but is extremely scarce. According to Huang and Carley [9], only about 2% of all tweets are geotagged. They also find that geotagging behavior changes significantly between communities based on language, country and other factors. Tweet location is extremely useful for various purposes, such as monitoring disease spread and natural disasters. Uses for tweet location data are plentiful, also including tourism information, local topic identification, emergency localization and local event recommendations [20]. There is a relation between user location and their tweet location which can be used to infer one from the other to some extent.

Mentioned location - Some tweets can mention places they are referencing. These locations can often be completely unrelated to the tweet location, such as when a user tweets about a place they visited in the past. Distinguishing mentioned locations from tweet locations is difficult but there are language processing techniques which can separate between the two to some extent. One example of this is the spaCy library used by Serere *et al.* [17]. The disambiguation of mentioned location and tweet location is important to improve the accuracy of inference methods since these two location types can interact with each other and often be mistaken for each other.

4 Implications of location inference

The lack of rules regarding what can and can not be shared on social media has lead a portion of users to feel like any and all aspects of their life should be shared. The reasons each user has for this can vary, but the fact remains that many people reveal information about their location, sometimes unknowingly, on a daily basis. Faced with this opportunity, many researchers wondered if it is possible to sift through the onslaught of data in order to find important information about events around the world.

Sakaki *et al.* [16] observed that many people tweet about an earthquake when such an event occurs. Their approach was to consider users as sensors, and use their location to determine the center and trajectory of the event location. The solution was put in practice in Japan, which was chosen due to its high earthquake frequency and high number of Twitter users. In this way, they were able to use the tweets of Japanese users to quickly announce the presence of earthquakes, with a reaction time faster than the Japan Meteorological Agency (JMA).

Similarly, Gomide *et al.* [8] were able to monitor the evolution of the Dengue epidemic in tropical and subtropical regions of Brazil. They did this by using the tweets of users and the spatio-temporal information linked to them. They were able to do this by analysing the correlation between the number of patients and the number of Dengue related tweets in the same time window, among other things.

Both of these examples were only achievable due to the fact that users share updates and because location information for these updates is available. For updates missing location information, location inference techniques are necessary to be able to use them. Presently, many researchers continue to work on improving the accuracy of location inference using Twitter, arguing that better localization can lead to faster detection of the event source. The benefits are clearly understandable, as this can lead to faster response times and more data for future precautions. We elaborate more on state-of-the-art methods in Section 5.

However, it is also important to keep in mind the possibility that this publicly available information might be abused. For example, Sadilek *et al.* [15] show that it is possible to infer users' friend networks from their location and vice versa. Additionally, Li *et al.* [12] were able to infer the demographics (i.e. age, gender, education) of users simply by tracking their location information in a specific time window. Possibilities of privacy leaks such as these are what drive most experts to recommend that users hide their location information on all social media platforms [1]. It is important to highlight that these are just some examples of what can be extracted from social media

without the user's intention. There is a large potential for misuse of this kind of information. Inference approaches fall into this category as well by filling in gaps in data and essentially circumventing the users' unwillingness to share certain information about themselves.

In order to have a more in depth look at what users are aware of, Alrayes and Abdelmoty [5] carried out a survey. Among other questions, they asked the participants to react to several statements regarding what information can be inferred from their social media. For example, two of the statements were: "I can guess where your home is." and "I can guess when you are AWAY from home.". Figure 1 shows the compiled results of the study grouped by social media platform. This emphasizes that more than half of Twitter users, 55%, are not entirely comfortable with this lack of privacy. And the percentage of worried users is even higher for other platforms. It is important to note that this study was published in 2014 and that data collection, algorithm power and user behavior have changed significantly since then.

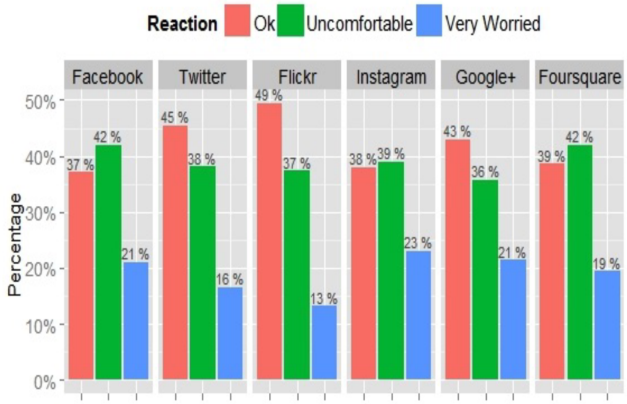


Fig. 1: Users' comfort level with the amount of information that can be inferred about them [5].

We can therefore conclude that the implications of location inference are vast and need to be carefully considered moving forward. There are clear advantages and benefits that can be observed when it comes to event localization. Authorities are able to react faster and coordinate better when information is abundant, and the public is able to make informed decisions. However, as methods for precise localization get more sophisticated, we approach the dangers of reduced personal privacy. Malicious actors may be able to exploit the availability of this kind of information with nefarious motives. It is important to carefully assess the situation as researchers are developing improvements in this growing field of research.

5 Methods

Location information can be of varying accuracy levels. It can be as broad as finding the country or as narrow as a Point of Interest (POI), which references a specific building or establishment. Later in this section we will mention several types of location inference techniques which have been developed, along with examples of each.

5.1 Input data

Twitter can be used to find an enormous amount of data about its users [18], such as demographics, age, social class etc. The basic information from which this data can be extracted is the user's following network, their tweet content and tweet context. The following network is a directed graph of follower relationships between users. Tweet content is only the actual textual content of the tweet. Audiovisual content such as images or video can also be included in this data but methods able to use this kind of data are rare. Tweet context refers to the context in which a tweet is posted, its metadata such as time of posting, whether it is replying to another tweet, etc.

Within each of these input categories, there are several methods of using data to infer one of the three location types [20]. Each of these inputs can be used differently. For example, the user network structure conforms best to graph-based analysis since it is inherently structured as a directional graph. Davis Jr. *et al.* [7] have shown that follower-follower relationships can be used to geolocate tweets.

5.2 Time based approaches

Home location and several other similar variations, such as work location, can be inferred from tweet locations and context. Huang *et al.* [10] uses a spatio-temporal clustering method to approximate users' daily habits from their tweet locations. These methods utilize the repetitive nature of human movement to pinpoint locations which users visit regularly. Then they use the Google Maps API and urban land use maps to extract the type of location and infer the activity being performed there by the user. This lets them form a map of users' daily habits including a time window for each location. An example of such a map is shown in Figure 2. We can observe the high quantity of tweets the user posts in one location, most likely their home, followed by a smaller quantity of tweets at a secondary location, probably their workplace. The other spikes represent other areas that the user frequents often. The specific details of which location belongs to which category can be found using land use maps and other means. They are not shown in Figure 2 as it is purely a visualization of tweet frequency per location.

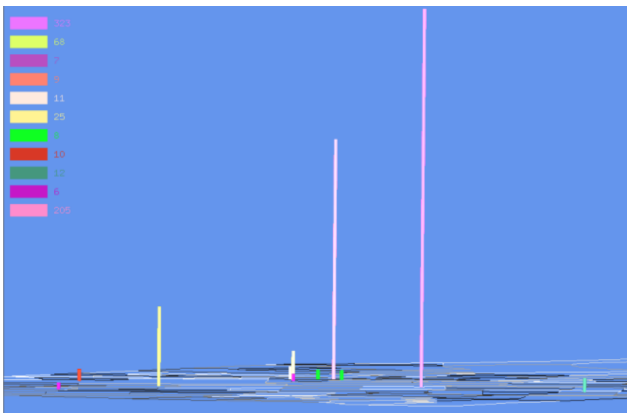


Fig. 2: Geo-tagged footprint distribution of a Twitter user in St. Louis, Missouri [10]

Other approaches include modelling tweet frequency for different POIs over time and using this information along with tweet content to infer the tweet location on a POI level. It can be found that using a big enough training data set, possibly expanded by searching the internet for keywords for each POI, this method can predict tweet locations quite accurately [13]. This is yet another approach which can deduce tweet location based on content and context only.

5.3 Network based approaches

Multiple ways to use the networking aspect of Twitter for location inference purposes exist. Sadilek *et al.* [15] presents a method for inferring users' networks from geotagged tweets and inferring users' locations from their friend network. They show that the combination of users' location, content of their tweets and friendship formation is a strong predictor of users' friendship status and vice versa. This is done using both supervised and unsupervised methods. These, once again, show the dangers of geotagging tweets en masse and various uses for such a dataset.

A user's network can also be used to identify their home location. Wagenseller *et al.* [19] use a supervised machine learning model to assign users without a specified home location to communities of users with known home locations. They use a Generalized Additive Model (GAM) due to its ability to highlight significant features. This model

allows them to pick the most likely community a user belongs to and to identify the user's home location using the location of the other members of the community. This use case is just one example of how novel machine learning methods and deep neural networks can be used to further improve location inference.

Kong *et al.* [11] finds that users' networks can infer their location quite well even without any specific algorithms. They discovered that 83% of individuals whose shared friends account for at least half of their social circle reside within a 10km radius of each other. This information in combination with other techniques mentioned in this paper can form a credible basis to infer locations of Twitter users based on their networks with fairly high accuracy.

5.4 Content based approaches

Since tweets are collections of words, they can be analyzed using Information Retrieval (IR) techniques. Several research papers propose an IR-based approach to finding a so-called 'locality' measure of words and identifying words which are local to some areas. Examples of such words can be colloquialisms like "howdy" (greeting popular in the southern United States) and "phillies" (citizens of Philadelphia) [20]. An analog to Inverse Document Frequency called the Inverse Location Frequency (ILF) is proposed to express this measure [14]. ILF assumes that local words appear in fewer locations, hence having a higher ILF score. This can then be used to estimate the home location of a user given their tweets following a statistical distribution.

There are many approaches which are not mentioned in this section, as it simply serves as an overview of the possibilities in regard to location inference on Twitter. There is a lot of research in this area from both recent and older studies. If one were to combine all proposed methodologies and create an ensemble of Twitter location inference techniques, it is likely they could surpass the capabilities of each individual technique and provide extremely accurate data. The availability of such methods can be endangering to some individuals who may be unaware of the predictive power of such techniques, as we have mentioned in Section 4.

6 Exploratory study: LLM based location inference

So far, in this paper we have looked at the current state of the art approaches used to determine the location from which a tweet might have originated. Since this is an ever-expanding field of research, we decided to investigate if the recent advances in Large Language Models (LLMs) might contribute to the available solutions. The inspiration behind this idea is the fact that a recurring problem for current architectures is the grammar used by users. As elaborated in Section 3, tweets have a limited amount of characters, so users often use abbreviations, slang and emoticons. Since OpenAI's GPT-3.5 model, like all current LLMs, was trained on such data, we expect it to be a good solution for analyzing tweets. In order to verify this, we carried out an empirical study on a dataset of tweets posted by users whose location is known. We gave all the tweets of one user to the ChatGPT model and asked it to guess where the user might be located. Our results show that LLMs could definitely be used to extract location information from posts. However, we were not able to localize all users, so we conclude that LLMs would best be used in combination with another algorithm currently available.

In the following subsections, we elaborate on the dataset, methodology and results of our empirical study.

6.1 Dataset

For our empirical study we used a January 2010 Twitter scrape dataset that we found through a GitHub repository listing publicly available Twitter datasets [3]. The dataset was created by Z. Cheng *et al.* in order to build a framework that can locate users based solely on their tweet contents. Z. Cheng *et al.* describe the dataset and their findings in 'You Are Where You Tweet: A Content-Based Approach to Geolocating Twitter Users' [6]. The dataset contains both a training and a testing set, but since we did not train our own model, we only used entries from the training set, as those had a structure which matched our needs better. The training set that was used contains approximately 3.8 million tweets collected from 115,886 users. For each user, the dataset contains their self-reported location on the city level. The dataset does not contain users without a location, since it is designed for location inference. Since access to the OpenAI REST API is paid based on the amount of tokens sent, we used 1000 randomly selected users for our empirical study. This means we used 47,919 tweets, which averages out to around 48 tweets per user.

6.2 Methodology

Once we obtained out dataset, we used the OpenAI Chat Completions REST API to query the ChatGPT model `gpt-3.5-turbo-0125` (accessed on 25 Feb 2024) to guess the location of users based on their tweets only. We used this method to obtain a proof of concept showing that LLM-based location inference techniques are a viable solution. Therefore we report the findings as motivation to include LLM-based analysis in future location inference methods.

We used a system prompt, which informs the model about how it should respond and what attitude to have towards the conversation. We set this system prompt to limit the amount of words used in the responses, since we wanted the answers to just contain location guesses and nothing else. The system prompt used was *"You only use maximum 10 words when responding."*

To give the actual tweets to the model, we used a framing prompt to explain the situation and to describe what answers we were looking for. This prompt was *"Following are the tweets of one Twitter user. Please try to guess the location of this Twitter user by saying what city and the country they might be from. You may try your 3 best guesses."*. We appended all the tweets we had for a user separated by newline characters to this message and sent the request through the API. The responses we got from the model varied in format, sometimes using a numbered list for the guesses, other times just containing a list of cities. However, we were successful in gathering only city guesses and no other information in all the responses.

In order to assess the responses we got from the system, we checked if the correct city was among the 3 guesses provided by ChatGPT. This was done by taking the city name and checking whether it was a substring of the model's response. Due to the unstructured nature of LLM responses we did not parse and score each of the three guesses. However in future work it should be possible to provide the model with a detailed response template in order to make its outputs parseable. This could be achieved by altering the system prompt and adding more detail to it. We did not go this far since this was only a small empirical study. For clarity, a diagram of the experimental process can be found in Figure 3.

It is important to note that if the LLM is unable to infer a location from the content provided, it can hallucinate and guess at random. It is highly unlikely that a random guess will be correct, so this did not affect the accuracy in our study. If using this for inference, this would become a problem which would need to be solved. We believe that prompt engineering could be used to avoid random guesses and to prevent the LLM from giving incorrect answers in case of uncertainty. The LLM can also admit that it doesn't know and give no guesses. In this case, the LLM could be instructed to reply with a specific sentence if it cannot infer the location using the system prompt. This standardized response could be parsed and used to distinguish users whose tweets do not contain enough information to be localized.

6.3 Results

After we gathered the guesses that ChatGPT made regarding the location of each user, we checked to see how many contained the correct city name. Out of the 1000 users, the model was able to correctly name the location of 478 of them. This means we obtained an accuracy of 47.8% in our empirical study. This accuracy is not as high as other existing methods, however it is still very good considering the number of possible results. The main finding of this study is not a stellar accuracy number, but the public availability of tools which achieve

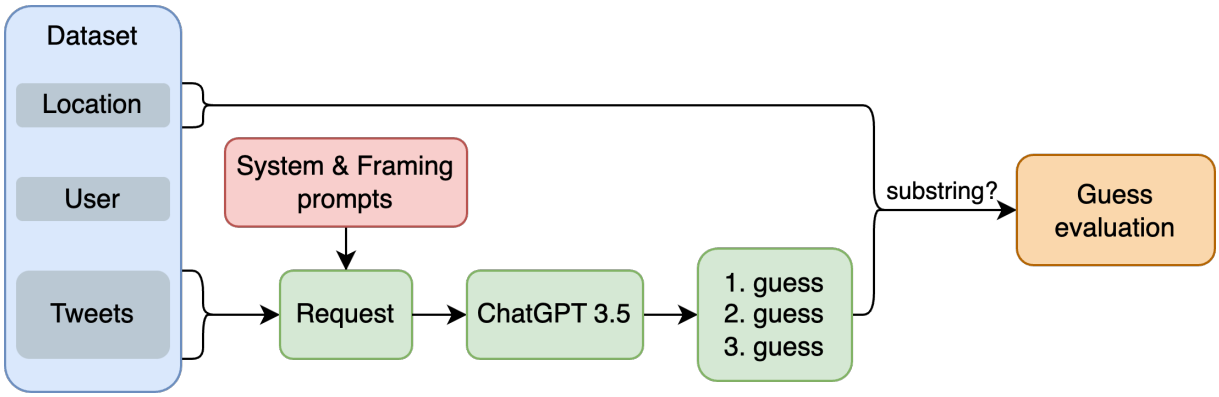


Fig. 3: Diagram of the experimental process.

decent accuracy. These results show that LLMs can indeed contribute in this field of study. They could be used to obtain a close estimation of the location, which another model could use as a starting point to get a more accurate and precise answer. We discuss the implications of these findings in Section 7.

7 Discussion

Here we discuss the validity of this paper. Several techniques which we cite in this paper are somewhat old relative to the fast-moving Internet trends and developments. We included some more recent studies such as the machine learning solution [19] and the data selection study [17]. However, many of the approaches mentioned are only from 2015 or earlier. There may be other approaches in the works which are not yet published but which may improve on the previously defined techniques in novel ways. It is also likely that Twitter has internal tools which perform better than most other approaches. This is because the platform likely stores even more information about its users compared to publicly available data.

The outcome of our exploratory study presents a unique dilemma. It shows that freely available tools like the GPT model family by OpenAI can serve as a powerful inference tool. This greatly reduces the barrier to entry for location inference techniques which required scientific understanding and the ability to reproduce methods described in scientific papers such as the ones we mention in Section 5. But with the availability of LLM-based tools it is possible for parties wishing to abuse location inference techniques to do so much easier. We wish to shed some light on this topic and conclude by bringing attention to the potential implications of this technology being readily available.

7.1 Drawbacks and limitations

As mentioned in Section 4, location inference on tweets can have beneficial purposes. However, such techniques can also suffer from drawbacks, such as the fact that Twitter datasets are hard to find. Changes to the Twitter API make it so that pulling data from the platform for free by anyone is now largely restricted [4]. Research accounts are available but still limited. Therefore the usage of such data for non-commercial purposes designed to benefit society may become a challenge due to the costs associated with gathering large datasets.

Data scarcity remains a problem for any technique as well. Naturally, location inference cannot be accurate when given very little input information. This is an inherent problem and will make it impossible to locate short and contentless tweets entirely. This limits the possible accuracy of any method due to the nature of the dataset.

Scaling inference methods to larger datasets remains a problem as well. Due to the nature of location inference, there is some uncertainty associated with each inferred value. If one was to use location inference on a large scale, the scarcity of ground truth geotagged data would quickly force the usage of inferred data as inputs to the next inference steps. This global inference comes with its own downsides as the propagation of incorrect labels may invalidate large parts of the dataset through increasing uncertainty and built-in bias. Once we look at such a big dataset, we also need to take into account that it is constantly changing. Users can delete their tweets or change their location. This presents another challenge for any approach which needs to be considered. It is unclear how this can be avoided in this specific use case.

8 Conclusion

The location information of an individual is an extremely private piece of information and can be misused by malicious actors. Therefore, it is an important factor of privacy on the Internet to be able to control whether one's location can be inferred from the contents of their internet presence. It is entirely possible that some users are comfortable with having their location be known to strangers, but this is not the case for all Internet users.

With that being said, location information can often be used for legitimate purposes. Location data can be used for natural disaster remedy and early detection, disease tracking and many other purposes. These are use cases which are beneficial to society and can be achieved without breaching people's privacy. They form an argument as to why location inference on social media is an important tool in a researcher's arsenal.

Location inference techniques presented in this paper, as well as many more which are not directly mentioned, achieve various levels of accuracy in their results. They can take various forms of input and infer several different location attributes related to users or their tweets. In combination, they may be able to infer much more information than each of them on their own. The addition of machine learning, deep neural networks and large language models can also make these techniques much stronger.

8.1 LLM-based methods

We show that the GPT-3.5 LLM can perform fairly well on its own to infer a user's location from their tweet content. As it is publicly available, it allows anyone with access to the Internet to perform location inference. It can solve many problems which previous techniques may have struggled with such as locality of words, disambiguation of meanings and distinguishing between tweet location and mentioned location with ease. LLMs may become a very useful tool in location inference in the future, especially if integrated with the plethora of other techniques which already achieve good results. However they may also become a dangerous tool in the hands of people acting in bad faith.

We would like to make a recommendation to users as to how they can avoid or decrease the likelihood of their location being inferred from their online presence. However, this proves quite difficult because, as we mention in Section 5, there are so many various approaches using different inputs that it is unclear whether such a recommendation can be given. Naturally, users can simply choose not to share any data online, but that is not a true solution. We have seen that

users' networks, their friends and even their posting frequency can be used for location inference. Hence, we can only recommend that users who wish to keep their location private abstain from disclosing it outright, and consider what they share and who they connect with as all of those may be factors which can help infer their location.

Overall, there is an abundance of techniques for location inference from social media posts. With the rise of LLMs and other powerful computational tools, the task may become even less daunting and the accuracy of future solutions will keep increasing. Datasets with inferred location may become available and may be used for various purposes in the future.

9 Future work

We propose several extensions of our work in this section.

Neural network based approaches - It could be interesting to investigate whether approaches using deep neural networks may be more effective at inferring locations due to their ability to find and compute relationships which statistical models may not find. A deep neural network model may be able to combine inputs which are already defined and being used in a novel way. With an appropriate level of preprocessing and a good architecture, an approach like this may perform quite well.

Validation across social media platforms - In theory, all approaches mentioned in this paper could be applied to other social media platforms with textual content. However, due to differences in culture and convention as well as different implementations, it may be somewhat of a challenge to adapt these techniques for optimal performance on other social media. It could be an interesting topic of research to see which parameters need to be adjusted in order for optimal performance on other social media. As an extension, even social media mainly focused on images instead of text could be analyzed since they usually have a description field for images which could be utilized as input for the algorithms created to work on textual input.

9.1 Exploratory study extensions

As future exploration related to our LLM study, we recommend checking how close the guesses were to the correct location. The reason for this being that we observed that sometimes the model guessed quite close to the actual location. For example, we saw in one case that the model guessed Miami while the correct location was Palm Beach, a neighboring city. It was out of the scope of our study to try and quantify the quality of guesses like these, so they were marked wrong. Another interesting investigation may be to compare various LLMs and their performance for this specific task.

Our study could be further extended by finding the best way to use an LLM in the location inference pipeline. There are multiple steps in the proposed pipelines for location inference and multiple of them could use an LLM as a helper tool. It could be used in decoding and parsing the textual content as shown in this paper. An LLM could also be used to quickly, effectively and accurately parse user-entered location data into a standardized format. There are libraries which are usually used for this step, but it could be interesting to investigate the performance of an LLM in this task. Furthermore, there are other tasks in the pipeline which could be simplified, split up or even verified using an LLM. We hypothesize that a combination of traditional methods and LLM-based methods could achieve performance superior to existing approaches if it takes advantage of the best of both worlds.

Acknowledgements

The authors wish to thank the Student Colloquium course at the University of Groningen and its coordinators for providing the opportunity to write this paper and publish it.
We also wish to thank the reviewers who helped us improve this paper, namely Dr. A. Rastogi, K. Vos and P. Teeninga.

References

[1] How to stay safe on social media. https://www.amnesty.org.au/how-to-stay-safe-on-social-media/, Nov. 2018. Accessed: 24-02-2024.

[2] One year in, the future of X is bright. https://blog.x.com/en_us/topics/company/2023/one-year-in, Oct. 2023. Accessed: 25-02-2024.

[3] awesome-twitter-data. https://github.com/shaypal5/awesome-twitter-data, 2024. Accessed: 26-02-2024.

[4] X API. https://developer.twitter.com/en/products/twitter-api, 2024. Accessed: 26-02-2024.

[5] F. S. Alrayes and A. I. Abdelmoty. No place to hide: a study of privacy concerns due to location sharing on geo-social networks. 2014.

[6] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10. ACM, Oct. 2010.

[7] C. Davis Jr, G. Pappa, D. Rennó Rocha de Oliveira, and F. Arcanjo. Inferring the location of twitter messages based on user relationships. *T. GIS*, 15:735–751, 12 2011.

[8] J. Gomide, A. Veloso, W. Meira, V. Almeida, F. Benevenuto, F. Ferraz, and M. Teixeira. Dengue surveillance based on a computational model of spatio-temporal locality of twitter. In *Proceedings of the 3rd International Web Science Conference*, WebSci '11. ACM, June 2011.

[9] B. Huang and K. M. Carley. A large-scale empirical study of geotagging behavior on twitter. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '19. ACM, Aug. 2019.

[10] Q. Huang, G. Cao, and C. Wang. From where do tweets originate?: a gis approach for user location inference. pages 1–8, 11 2014.

[11] L. Kong, Z. Liu, and Y. Huang. Spot: locating social media users based on social network context. *Proc. VLDB Endow.*, 7(13):1681–1684, aug 2014.

[12] H. Li, H. Zhu, S. Du, X. Liang, and X. Shen. Privacy leakage of location sharing in mobile social networks: Attacks and defense. *IEEE Transactions on Dependable and Secure Computing*, 15(4):646–660, July 2018.

[13] W. Li, P. Serdyukov, A. P. de Vries, C. Eickhoff, and M. Larson. The where in the tweet. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, New York, NY, USA, Oct. 2011. ACM.

[14] K. Ren, S. Zhang, and H. Lin. Where are you settling down: Geo-locating twitter users based on tweets and social networks. In *Asia Information Retrieval Symposium*, 2012.

[15] A. Sadilek, H. Kautz, and J. P. Bigham. Finding your friends and following them to where you are. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM'12. ACM, Feb. 2012.

[16] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 851–860, New York, NY, USA, 2010. Association for Computing Machinery.

[17] H. N. Serere, B. Resch, and C. R. Havas. Enhanced geocoding precision for location inference of tweet text using spacy, nominatim and google maps. a comparative analysis of the influence of data selection. *PLOS ONE*, 18(3):e0282942, Mar. 2023.

[18] L. Sloan, J. Morgan, P. Burnap, and M. Williams. Who tweets? deriving the demographic characteristics of age, occupation and social class from twitter user meta-data. *PLOS ONE*, 10(3):e0115545, Mar. 2015.

[19] P. Wagenseller, Y. Zhao, F. Wang, and A. Avram. Community-based location inference in social media using supervised learning approach. *Social Network Analysis and Mining*, 11, 12 2021.

[20] X. Zheng, J. Han, and A. Sun. A survey of location prediction on twitter. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1652–1671, 2018.

Unraveling the Architecture of Digital Twins: From Conceptual Foundations to the Web of Interconnected Realities

Mia Müller, Mohammed Nacer Lazrak

Abstract—Digital twins (DTs) have emerged as a pivotal component of emerging technologies, revolutionizing industries through their ability to create virtual replicas of physical entities and processes. Despite the growing significance of DTs, there yet has to exist a comprehensive definition of what a DT is, where and how it can be used, and in which direction its development will take us in the future. We will begin by comparing various definitions of DTs, synthesizing technical and conceptual perspectives to establish a comprehensive understanding of the term. Through this comparison, we aim to clarify the fundamental principles underlying DTs and distinguish them from other computing paradigms. Furthermore, we will delve into the current landscape of DT deployments across industries, showcasing their transformative potential in optimizing operations, enabling predictive maintenance, and fostering innovation. Through this analysis, we will identify emerging trends and challenges in the practical implementation of DTs. After thoroughly examining the landscape of DTs, our paper will review the concept of a Web of digital twins, where interconnected DTs facilitate seamless communication and data exchange within and across domains. By exploring the architectural underpinnings and implications of this interconnected network, we aim to provide valuable insights for future research and development in the field of digital twins.

1 Introduction

In today's increasingly complex and interconnected world, managing and optimizing physical systems pose significant challenges. Traditional approaches often fall short in providing real-time insights and a comprehensive understanding of these systems' behaviour, leading to inefficiencies and missed opportunities. However, emerging technologies offer a promising solution: digital twins (DTs). Digital twins address the challenge of efficiently managing and optimizing complex physical systems by providing a comprehensive representation of them. Traditional approaches to monitoring and managing such systems often lack visibility into their inner workings and real-time performance, leading to inefficiencies, costly downtime, and suboptimal decision-making. Digital twins bridge this gap by, for example, continuously collecting data from sensors and other sources, often enabling accurate modeling, simulation, and analysis of the physical counterpart's behaviour. By leveraging advanced technologies such as AI and IoT, digital twins might soon empower stakeholders to monitor, predict, and optimize the performance of assets, processes, and entire systems, ultimately driving improved operational efficiency, productivity, and innovation across various industries. Digital twins were initially introduced by Michael Grieves in 2002 [7] in the context of a conceptual model underlying product lifecycle management as a general description of digital twins. Digital twins are virtual representations of physical objects, systems, or processes. They are created using real-time data from sensors, devices, or other sources to track the physical counterpart's behaviour and status. Practically, digital twins are utilized in manufacturing to optimize production processes by simulating and analyzing equipment performance, reducing downtime, and improving efficiency. In healthcare, digital twins could soon be used to model resources, such as ambulances or operating rooms. Revolutionary uses of digital twins could involve creating personalized digital replicas of individuals, allowing for virtual simulations of medical treatments tailored to each person's unique physiology and health conditions. In urban planning, advanced digital twins could simulate entire cities' ecosystems, enabling predictive modeling of climate change impacts, resource management, and sustainable development strategies.

It is clear that digital twins have great potential, yet their novelty has also led to a multitude of un-unified research. Researchers have yet to agree on a comprehensive definition of digital twins, areas of implementation are vast, different architectures have been proposed, and yet researchers are already exploring the future of digital twins. In this paper we will create a point of reference, an overview of the topic, aiming to create a unifying starting point for any future research. We will follow the following research questions:

RQ1. What is the (minimal) definition of a digital twin?

RQ2. What can the architecture of a digital twin look like?

RQ3. What are future endeavors of implementing digital twins?

In the first section, we are going to give an overview of the current implementations of digital twins and explore the different definitions discussed in the literature, making a practical differentiation between them and bringing forward one in particular. The implementation of digital twins is complex, so after discussing the definitions, we will introduce the possible architecture of a digital twin on multiple levels, from concept modeling to deployment. Finally, we will explore a definition of the web of digital twins, some general design principles when attempting to make it, and a more concrete potential architecture.

2 Defining Digital Twins

The concept of digital twins has gained significant attention, yet its precise definition remains a subject of debate. To synthesize thorough and applicable definitions we must look at what has already been implemented under the term "digital twins" and how digital twins are defined in the literature, analyze whether and how implementations align with these definitions, and how/if these definitions overlap.

This section is therefore structured into three parts: first, an overview of current implementations will be given, next, current definitions will be introduced and finally both findings will be compared and a definition will be brought forth. This section will answer the first research question.

2.1 Current Implementations

The possibilities for digital twins are vast. To get a better overview of how and where digital twins are implemented, we will examine some real-life implementations in this section.

The city of Groningen boasts its own digital twin, a comprehensive digital replica encompassing the underground, ground level, and topsoil of the municipality. This digital twin showcases the city's buildings in a 3D format, providing valuable tools for enhanced design, citizen participation, communication, and decision-making in the planning of new city structures [10]. The team behind this ambitious

project aims to engage citizens in city planning processes and streamline communication among all stakeholders involved in major projects.

Updated with new city data every two years, this model enables a variety of analyses, such as sun and shade assessments, water distribution analyses, profile line evaluations, as well as 2D and 3D measurements. Another notable advantage highlighted by the team is the model's versatility in data download formats, utilizing OGC standards to facilitate data exchange with architectural firms, educational institutions, governmental bodies, and residents alike [10].

Beyond these benefits, the digital twin is anticipated to yield long-term cost savings by serving as a centralized data hub. This eliminates the need for data conversion, synchronization, and copying across different parties, along with mitigating associated risks of failure in these processes.

Digital twins are not constrained by size, as demonstrated by the National Digital Twin Programme (NDTP) in the United Kingdom. The NDTP, a government-led initiative, is dedicated to enhancing the country's capabilities in digital twinning technologies and processes [6].

Currently, the NDTP has distinct objectives. It aims to develop a digital twin capable of exchanging relevant information, adapting and evolving over time in a sustainable manner, and ensuring safety, security, and trustworthiness while keeping ethical considerations in mind. This project delves into how a digital twin can assist in assessing the feasibility and outcomes of policies, as well as preemptively planning for unforeseen events.

A final example of a digital twin on a more localized scale is GE Renewable Energy's digital wind farm project. The primary objective of this endeavor is to leverage data obtained from actual wind farms to model the intricate interactions between turbines, wind patterns, and terrain. Through this process, a detailed analysis to determine the ideal turbine placement for each location is made possible. Implementing this digital twin technology for GE Renewable Energy's wind farms is projected to boost energy production by up to 20%, translating to an estimated additional revenue of $100 million for the company [5].

2.2 Current Definitions

The examples shared earlier show that digital twins fundamentally involve modeling a physical asset and often include additional functions. For instance, the digital twin of Groningen allows for sun- and shade analysis. However, defining this concept precisely is challenging.

Researchers at NASA defined DTs as "real-time multi-scale multi-physics simulations" of physical systems, designed to predict system failures in real-time [13]. The definition is quite narrow, it requires the system to run in real-time and asks of a digital twin to be capable of predicting system failures. It also requires a digital twin to be a representation of a physical entity. This somewhat aligns with what we have seen in the examples of implementation but also excludes some. For example, the digital twin of Groningen is only updated every two years and does not aim to predict failures.

Other definitions split the advanced capabilities of e.g. failure prediction from the monitoring aspect of a DT. This paper [2] introduces the concept of the digital shadow (DS) as a part of a digital twin. The digital shadow is the digital representation of a corresponding physical asset that is updated continuously through e.g. sensor data. The DT then extends the DS, enabling some form of functionality that exceeds that of simple modeling, for example, error prediction. This definition offers greater precision regarding the essence and functions of a digital twin. However, were it implemented as a universal definition, some projects may have to be reclassified as digital shadows. Moreover, it emphasizes the necessity for unique functionalities within digital twins, possibly raising questions about the benefit of imposing such restrictions on the concept. This split is also in direct contrast to other definitions that define DTs as the combination of physical and virtual systems where changes in one are reflected in the other [12].

Yet another definition defines DTs as "real-time decision-support sandboxes" [1], focusing on a DTs ability to create an isolated space to e.g. test new solutions (like building new buildings in Groningen [10]) and by doing so aiding in the decision-making process.

Moving towards distributed systems, DTs could be defined to offer a means to provide complete semantic descriptions of assets, leading to enhanced business processes and advancements in smart manufacturing [9]. This integration with distributed systems also introduces the potential for a collaborative nature of DTs in modern industrial landscapes.

2.3 Definition of Digital Twins

In summary, the term "digital twin" as of right now is more of an umbrella term rather than a precise technical concept. Some core aspects reappear in all we have seen though, implementation and definition alike. A digital twin always represents a physical asset, which is not specified in size or domain and the digital twin must be continuously updated with real-life data to accurately represent its physical counterpart. Differences appear when we take a closer look. For example, some definitions show that the digital twin must be updated in real time, but we have also shown implementations where updates occur only every two years. Other disagreements seem to be whether the modeling aspect of a digital twin should be its own concept, i.e. a digital shadow, or whether this is simply a core part of a digital twin or if it is the main functionality of a digital twin. Finally, there is no clear agreement on if and which additional functionality a digital twin should have (e.g. error prediction, data sharing, etc.). As of right now, the precise definition is left to the individual. Taking all of these aspects into consideration, we have defined digital twins as follows:

Definition. *A digital twin is an integrated, real-time digital replica of a physical entity, closely mimicking its physical counterpart. A DT must be updated continuously through real-life data to accurately represent its physical counterpart, where the frequency at which the DT is supplied with real-world data is system-dependent. A digital twin may have the ability to aid in prediction, decision-making, virtualization of data, data analysis, and data exchange, but does not require such functionality.*

3 Architecture of Digital Twins

Just as there are diverse definitions for digital twins, there are also multiple proposed architectures. To date, no unified, foundational architecture has been established.

The importance of a standard architecture cannot be overstated, as it would significantly broaden access to digital twins, particularly for businesses that lack the resources to develop their own comprehensive models. Given that digital twins are a crucial element in the EU's transition to Industry 4.0 [3] , the creation of a standard is imperative for their widespread adoption. Moreover, this standardization would pave the way for potential interoperability, promising substantial impacts as discussed in section 4.

This section aims to consolidate key elements of digital twin implementation into an architectural guideline. It will explore three pivotal areas, as outlined in three separate papers: logic components, communication patterns, and deployment and infrastructure.

3.1 Concept Modeling and Logical Components

The first architecture we will present is proposed by Alessandro Ricci *et al.* [11]. Here, the concept modeling and logical component architecture of a digital twin are presented in the context of healthcare with examples of DTs include ambulances, operating rooms, etc.

3.1.1 Modeling a Physical Asset

The main component of any digital twin is its ability to model a physical asset (PA). Before a PA can be modeled well, we must take a closer look at what aspects of the PA are important. This paper introduces the following components that must be taken into consideration:

- **Properties:** Observable attributes of the PA, depicted as labeled data values dynamically changing with the PA's state evolution.
  - **Example:** The location of an ambulance.

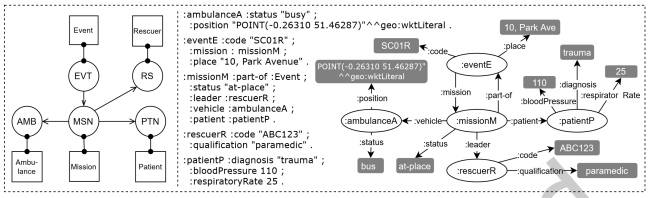


Fig. 1. An example of a RDF-based Knowledge Graph (on the right) for a WoDT (on the left) in the healthcare context.[11]

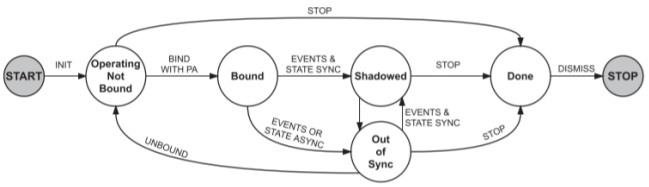


Fig. 2. Abstract representation of the state and transitions of a DT[11]

- **Relationships:** Connections between the PA and other PAs, serving as links to other digital twins. They can be observable, dynamically created, and subject to change over time, extending beyond the local PA state.
  - **Example:** A relationship to (the digital twin of) the driver operating the ambulance.
- **Events:** Observable occurrences at the PA level, reflecting significant events within the domain.
  - **Example:** The ambulance has low fuel.

3.1.2 State Modelling

Now that we know what to focus on regarding the PA, we can take a look at what is important about a digital twin. The paper introduces *states* that any digital twin goes through in relation to its corresponding PA. There are five states a digital twin can be in, namely:

1. **Operating Not Bound:** The digital twin has been initiated but is not yet bound to its PA.
2. **Bound:** The digital twin is bound to its PA.
3. **Shadowed:** The digital twin is a correct representation of its PA.
4. **Out of Sync:** The digital twin is an outdated representation of its PA.
5. **Done:** The digital twin is no longer needed and will be shut down.

A digital twin will always be at exactly one state at a time during its lifecycle. Please refer to figure 2 to see the events triggering a change in state.

3.1.3 Logical Architecture

With a clear view of the modeling of a PA and which states a DT should move through during its lifecycle, the paper is now able to propose an architecture for DTs. It is made up of multiple components as can be seen in figure 3.

- **Physical Asset Adapter:** Captures events from Physical Assets (PAs) and ensures uniform representation for seamless exchange. Handles both one-time binding and perpetual synchronization processes.
- **Binding & Shadowing Module:** Facilitates DT-PA association and perpetual synchronization. Manages DT lifecycle stages.

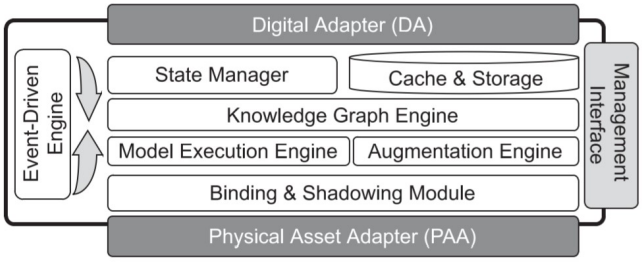


Fig. 3. Abstract Architecture of DT, excerpt of a graphic taken from [11]

- **Event-driven Engine:** Enables interaction among internal components through events.
- **Model Execution Engine:** Governs DT components based on predefined models, dictating event capture and state influences.
- **State Manager:** Ensures consistent DT state updates aligned with captured events and contextual conditions.
- **Knowledge Graph Engine:** Manages the DT's Knowledge Graph and provides caching and storage functionalities.
- **Cache & Storage:** Offers basic storage and caching functionalities for DT-related data.
- **Management Interface:** Exposes functions for DT management tasks, including lifecycle state queries and linking requests.
- **Digital Adapter:** Translates events for interaction with external entities and facilitates notifications.
- **Augmentation Engine:** Extends DT capabilities through functional modules, enabling additional properties and actions.

3.2 Component Communication and Data Store

As discussed in section 2, AboElHassan *et al.* [2] separates a digital twin into a digital shadow and a digital twin. Though we have chosen not to follow this definition, their proposed architecture is well structured, regardless. For clarity's sake, we will refer to the digital shadow in this section to explain the proposed architecture, though under our definition all that is part of the digital shadow is simply part of the digital twin. Additionally, keep in mind that under this paper's definition of a digital twin, a digital twin will always offer more functionality than just a virtual model of a physical asset, which is clearly reflected in the proposed functionality and architecture. As shown in figure 4, a digital twin following this architecture is made up of six functionality components:

1. Data Stream Processing
2. Data Lake
3. Real-Time Digital Model
4. Simulation Models
5. Recommendation System
6. Business Intelligence

Where the first three are further classified as part of a digital shadow. Possibly more interesting is the proposed architecture that can be seen in figure 5.

The architecture follows distributed system concepts, making use of the benefits a microservice architecture provides, such as independent management of each component. Here, what is seen in figure 5 under digital shadow, are all components needed to model the physical asset. The DT Sandbox shows all other, additional functionality, such as prediction models. Under our definition, this would be an optional component to implement when building a digital twin. Precisely, the three main components offer the following functionality:

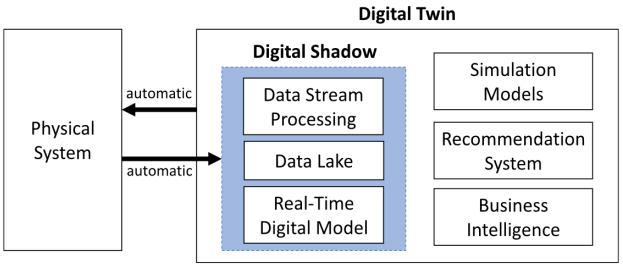


Fig. 4. Functionality-based separation between digital twin vs. digital shadow[2]

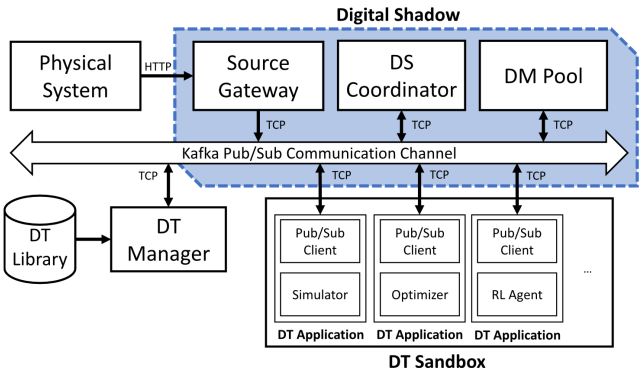


Fig. 5. Digital twin framework architecture[2]

- **DT Manager:** Responsible for administering the Digital Twin (DT) framework. It oversees deployment of DT applications using scripts and configuration files stored in the DT task library. It controls the Kubernetes cluster for orchestrating DT component deployment, defining configurations, managing resources, and ensuring smooth operation of the DT system.
- **Pub/Sub Communication Channel:** it facilitates asynchronous communication between DT components. Utilizes Apache Kafka, a high-throughput, low-latency messaging system, for distributing messages published by DT components. Kafka brokers, deployed on Kubernetes nodes, store message replicas and ensure fault tolerance. Topics in Kafka messages combine publisher and attribute names, enabling targeted message subscription.
- **Digital Shadow (DS):** It creates virtual replicas of physical systems. Collects data from physical system sources (e.g., sensors or devices), publishes on Pub/Sub communication channel. DS models, deployed in the DS Pool, use this data to simulate physical system behavior. Digital Model Manager (DMM) manages DS models, ensuring synchronization with real-time data updates and facilitating communication with other DT components.

The main advantage of this architecture is the loosely coupled components. Though this has not been proposed in the paper, we see the potential to offer (some) components as a service, possibly opening the doors of digital twins to smaller companies. This is not a novel idea, as shown in the next section.

3.3 Infrastructure and Deployment

Constantini *et al.* [4] introduces IOTwins. Funded by the EU, it directly targets SMEs as they often lack the resources to implement their own digital twin from scratch. Its goal is to create a platform that allows its users to "develop, configure and run DTs in the cloud-to-things continuum" [4].

Due to their platform being implemented on the cloud, its concept requires a different architecture to the one previously introduced. In the paper, three levels of architecture are specified, each can be accessed by an application layer with different levels of abstraction of data. The three layers are (1.) the IoT layer for Anomaly detection and control optimization, (2.) the edge layer for IoT management and orchestration, historical data collection, and ML model inference, and (3.) the cloud layer for simulation models, big data analysis, data retention and archiving, and ML model training (see figure 6).

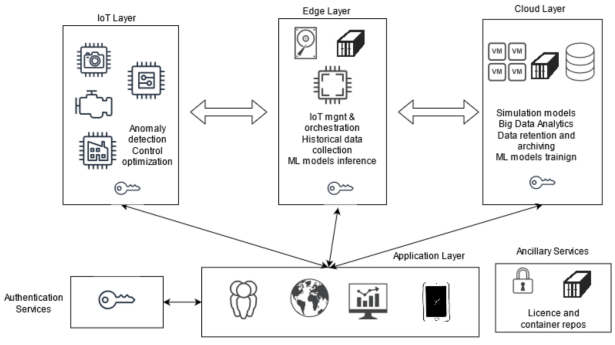


Fig. 6. High Level Architecture of IoTwins Framework [4]

This architecture aims to incorporate three requirements of DTs:

- Facilitate data transfer across infrastructure levels while considering real-time and non-real-time computation needs.
- Accommodate various data types and storage requirements, including short- and long-term storage.
- Enable both streamed data and batch-data elaboration, supporting on-the-fly and at-rest data processing.

3.4 Integration of Architecture Layers for Digital Twins

The creation of a comprehensive architecture for digital twins requires integrating the key elements from various proposed architectures. By synthesizing the logic components, communication patterns, and infrastructure considerations outlined in different papers, we can develop a robust framework for digital twin implementation.

3.4.1 Concept Modeling and Logical Components

A crucial aspect of digital twin architecture is the logical representation of physical assets and their corresponding digital counterparts. Drawing from Alessandro Ricci *et al.*'s proposal, the modeling of physical assets involves capturing properties, defining relationships, and identifying events pertinent to the asset's behavior. This logical representation forms the foundation upon which the digital twin operates, enabling real-time monitoring, analysis, and decision-making.

3.4.2 Component Communication and Data Store

Building upon AboElHassan *et al.*'s architecture, effective communication between digital twin components and robust data management are essential. Leveraging distributed system concepts and microservice architecture, the digital twin framework should incorporate mechanisms for asynchronous communication, such as Pub/Sub channels, ensuring seamless interaction among disparate components. Additionally, the architecture should encompass data processing, storage, and analytics capabilities to handle diverse data streams and support real-time insights and historical analysis.

3.4.3 Infrastructure and Deployment

Considering Constantini *et al.*'s approach, deploying digital twins in a scalable and flexible infrastructure is paramount. Cloud-based platforms offer the agility and resources necessary for hosting digital twin

applications across the IoT continuum. By adopting a multi-layered architecture encompassing IoT, edge, and cloud layers, digital twins can efficiently manage data flow, processing, and storage requirements while accommodating varying computational needs and scalability demands.

3.4.4 Holistic Architecture

Integrating these layers and components yields a holistic architecture for digital twins that addresses modeling, communication, data management, and infrastructure considerations. At its core, the architecture enables the creation, operation, and evolution of digital twins across diverse domains, ranging from healthcare to industrial automation. By adhering to standardized principles and leveraging emerging technologies, this architecture lays the groundwork for interoperability, scalability, and widespread adoption of digital twins in the era of Industry 4.0.

This integrated approach synthesizes the strengths of individual architectures to provide a comprehensive blueprint for implementing digital twins across various industries and applications.

4 Web of Digital Twins

The introduced architectural guidelines enable the implementation of digital twins in a standard fashion, facilitating the interoperability between different DTs. At this point, there is no doubt that digital twins pose great potential across various domains, yet their scope can extend even further. The concept of a Web of Digital Twins (WoDT) is introduced in Ricci *et al.* [11].

The authors present the WoDT as an open, distributed, and dynamic ecosystem consisting of interconnected digital twins. This ecosystem functions as an interoperable, service-oriented layer supporting applications, particularly smart applications and multiagent systems. Within a company, a WoDT can serve as a "service-oriented software layer on top of which smart applications can be designed and integrated, exploiting functionalities to access and interact with the interrelated physical assets as-a-service." [11]. The primary objective of such a web is to establish a context wherein multiple DTs can seamlessly interact with one another. For instance, in a hospital setting, each ambulance, doctor, patient could have their own DT, all interconnected to digitally represent the hospital as a whole. In essence, a WoDT acts as a blueprint to conceptualize the ecosystem from a computational standpoint [11].

Taking the first step towards such a complex system, design principles, an abstract model, and architecture designed to encapsulate the WoDT concept's fundamental aspects are introduced, independent of specific application domains or technologies.

4.1 Design Principles for Shared Digital Twins

In response to the challenges posed by the lack of standards and generalization in the field of digital twins, a set of design principles has been formulated to guide their development with a focus on usability in the WoDT. These principles emerged from interviews conducted with field experts [8]. These principles can be applied in combination with the described architecture guidelines above. The proposed design principles are as follows:

4.1.1 Data Link

A fundamental principle in DT design is the establishment of bidirectional data link capabilities. This ensures a seamless flow of data between the digital twin and its counterpart, facilitating enrichment with critical process parameters necessary for sharing and remote control.

4.1.2 Purpose

Customized functionalities are essential to equip the digital twin with the capability to process, transfer, and store data. These functionalities are tailored to the specific purpose of the digital twin, which is to enable cross-company and multilateral data sharing.

4.1.3 Interface

Interfaces play a pivotal role in enabling user interaction with the digital twin and facilitating direct, human-independent communication between distributed systems. This is crucial for enabling cross-company and multilateral data sharing.

4.1.4 Synchronization

Synchronization functionalities are designed to provide users with both real-time updates of incoming data and on-demand access to non-real-time data updates. This ensures that users have access to the most relevant and up-to-date information.

4.1.5 Data Input

The digital twin must possess the capability to process both raw and processed data, ensuring a comprehensive dataset of its counterpart. This completeness is essential for enabling cross-company and multilateral data sharing.

4.1.6 Interoperability

Interoperable interfaces are crucial for ensuring seamless communication between distributed digital twins. This includes semantic translation capabilities to harmonize diverse datasets, thereby enabling the sharing of interpretable and logical data sets across companies.

4.1.7 Data Security

Security concepts are integrated into the digital twin design to enforce usage control policies, allowing users to retain sovereignty over shared data. This is imperative for maintaining trust and enabling cross-company and multilateral data sharing while safeguarding sensitive information.

4.2 Architecture of Web of Digital Twins

Alessandro Ricci *et al.* describe the WoDT as a blueprint for conceptualizing an ecosystem from a computational standpoint [11]. This concept is reflected in the architectures presented in the paper, providing a general overview. The paper introduces two main types of architectures, along with an overview of interaction flow.

Central to the interaction flow is the notion of a digital adapter positioned in front of the digital twin, facilitating communication with it. In future implementations, such interfaces would require a level of generalization to ensure interoperability is maintained.

With this general interface of a DT in mind, let's delve into the details of the two proposed architectures.

4.2.1 Agentification of the Virtual Environment

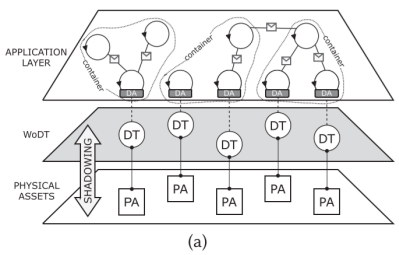
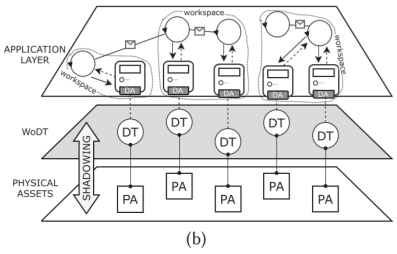


Fig. 7. Example of integration approaches using JADE platforms where agents wrap digital adapters (DAs) to mediate interaction with the DTs; figure from "Web of Digital Twins" [11]

The first approach involves the agentification of the virtual environment, as illustrated in Figure 7, where each digital twin within the Multi-Agent System (MAS) is portrayed by an agent serving as its representative or proxy. These agents provide an interface utilizing the Agent Communication Language (ACL), such as the FIPA ACL based on speech acts, enabling other agents within the MAS to interact with the DT.



(b)

Fig. 8. Examples of integration approaches using JaCaMo platforms where DAs become artifacts; figure from "Web of Digital Twins" [11]

In the second approach, Alessandro Ricci *et al.* propose the creation of an environment based on first-class environment abstractions, depicted in Figure 8 [11]. In this setup, DTs are represented as artifacts. Agents engage with digital twins by perceiving their observable states and events, and executing actions through interactions with these artifacts.

4.3 Important Findings in Web of Digital Twins

The exploration of the Web of Digital Twins (WoDT) concept sheds light on several crucial findings that underline its significance and potential impact across various domains. These findings are instrumental in understanding the implications and applications of WoDT in real-world scenarios:

1. **Interoperability as a Cornerstone**: The WoDT concept emphasizes the necessity of interoperability among digital twins, highlighting it as a cornerstone for effective communication and collaboration within the ecosystem. Interoperability ensures seamless interaction between diverse digital entities, enabling comprehensive data sharing and utilization.

2. **Service-Oriented Architecture for Smart Applications**: WoDT advocates for a service-oriented architecture (SOA) to facilitate the development and integration of smart applications. By providing a standardized software layer, WoDT enables the design and deployment of intelligent applications that leverage interconnected digital twins to enhance efficiency and functionality.

3. **Ecosystem Representation through Interconnected Digital Twins**: WoDT presents a holistic approach to ecosystem representation by interconnecting digital twins associated with various entities and assets. This interconnectedness allows for the creation of digital representations that mirror real-world environments, offering valuable insights and facilitating informed decision-making.

4. **Blueprint for Computational Ecosystems**: WoDT serves as a blueprint for conceptualizing computational ecosystems, providing guidelines and frameworks for designing and implementing interconnected digital twin environments. This computational perspective offers valuable insights into the architecture and dynamics of complex systems, paving the way for innovative solutions and applications.

In summary, the exploration of WoDT illuminates key findings regarding interoperability, service-oriented architecture, ecosystem representation, and the role of digital twins in complex environments. These findings underscore the transformative potential of WoDT across various domains, shaping the future of digitalization and intelligent systems integration.

5 Conclusion

In summary, this paper has examined the concept of digital twins, focusing on three aspects.

Firstly, regarding the definition of digital twins, it has become clear that while there are commonalities such as real-time replication and continuous data updating, there are variations in how these are implemented and the additional functionalities they may include. The proposed definition encapsulates digital twins as dynamic replicas of physical entities, adaptable to different system needs and capable of supporting decision-making processes.

Regarding the architecture of digital twins, our synthesis of various proposals has resulted in a framework. This framework emphasizes logical modeling, efficient communication, robust data management, and scalable infrastructure. It provides guidance on how to integrate physical assets with their digital counterparts, enabling real-time monitoring, analysis, and predictive capabilities.

Furthermore, our exploration of the Web of Digital Twins (WoDT) concept has revealed insights into interoperability, service-oriented architecture, ecosystem representation, and computational ecosystems. These insights underscore the potential of WoDT in fostering collaboration and enhancing system intelligence.

References

[1] A. AboElHassan, A. Sakr, and S. Yacout. *A Framework for Digital Twin Deployment in Production Systems*, pages 145–152. 01 2021.

[2] A. AboElHassan, A. H. Sakr, and S. Yacout. General purpose digital twin framework using digital shadow and distributed system concepts. *Computers Industrial Engineering*, 183:109534, 2023.

[3] CORDIS. Bringing digital twins to the edge for mass industry 4.0 applications, doi 10.3030/946009.

[4] A. Costantini, G. Di Modica, J. C. Ahouangonou, D. C. Duma, B. Martelli, M. Galletti, M. Antonacci, D. Nehls, P. Bellavista, C. Delamarre, and D. Cesini. Iotwins: Toward implementation of distributed digital twins in industry 4.0 settings. *Computers*, 11(5), 2022.

[5] G. R. Energy. A breakdown of the digital wind farm.

[6] D. for Business and G. Trade. National digital twin programme (ndtp), 2023.

[7] M. Grieves. *Virtually Intelligent Product Systems: Digital and Physical Twins*, pages 175–200. 07 2019.

[8] H. Haße, H. van der Valk, F. Möller, and B. Otto. Design principles for shared digital twins in distributed systems. 64(6):751–772.

[9] H. Haße, H. van der Valk, F. Möller, and B. Otto. Design principles for shared digital twins in distributed systems. *Business Information Systems Engineering*, 64, 04 2022.

[10] G. Leontien Spoelstra en Michel Grothe. Stad groningen in 3d, 2022.

[11] A. Ricci, A. Croatti, S. Mariani, S. Montagna, and M. Picone. Web of digital twins. *ACM Trans. Internet Technol.*, 22(4), nov 2022.

[12] R. Rosen, G. von Wichert, G. Lo, and K. D. Bettenhausen. About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine*, 48(3):567–572, 2015. 15th IFAC Symposium onInformation Control Problems inManufacturing.

[13] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang. Modeling, simulation, information technology and processing roadmap. 05 2010.

6 Appendix

In this paper we made use of the free version of ChatGPT (3.5). Prompts were some variation of "Please rewrite the following". In some sections we used bullet points and had ChatGPT rewrite it into one coherent text, however, this method proved to need too much rewriting on our part such that we did not implement this strategy any further. We paid especially close attention to quotes and citations when giving texts to be rewritten, as ChatGPT tends to rewrite quotes, too, and sometimes adds the wrong citations.

Self-Supervision with Graph Neural Networks
A Comparative Analysis of Generative and Contrastive Self-Supervised Learning

Maria Carmen Jica, Chan Chan Tran

Abstract—Self-Supervised Learning (SSL) tackles the issue of limited labeled data in real-world unstructured datasets by harnessing the inherent structure of the data. Graph Neural Networks (GNNs), specialized architectures designed for graph-structured data, can benefit from SSL by accommodating a broader range of training data by eliminating the need for ground-truth labels. SSL encompasses two primary approaches: generative and contrastive. Generative methods, like Autoencoders (AEs), strive to reconstruct input data, while contrastive learning maximizes the mutual information between different views of the graph. In this paper, we present GraphMAE, a novel model that combines SSL with a generative AE framework tailored for graphs, against the recent advancement of MaskGAE, a contrastive SSL applied in Variational AE. Our study conducts a comparative analysis between these two paradigms, aiming to critically assess their strengths and weaknesses, including factors such as structural robustness and real-world performance across diverse datasets. Through these experiments, we aim to elucidate the nuanced advantages and limitations of both techniques, offering insights into their optimal use cases and applications, and informing future research and practical implementation of graph SSL in GNNs.

Index Terms—Graph self-supervised learning, graph neural networks, graph representation learning, masked graph modelling, encoder, decoder, autoencoder, variational graph autoencoder, generative, contrastive.

1 Introduction

Graph Self-Supervised Learning (SSL) has experienced considerable traction because it enables the utilization of abundant unlabeled datasets from real-world contexts, thereby transforming them into valuable resources with minimal effort and expenditure. Graph Neural Networks (GNNs), specialized neural network architectures designed to process and extract features from graph-structured data, can adapt the SSL approach by capitalizing on the inherent structure of graph data, aimed at learning archetypal representations of nodes and edges without the need for external labels.

Within the realm of SSL, two principal sub-categories have emerged prominently: generative and contrastive. Generative learning, exemplified by Autoencoders (AEs), focuses on reconstructing the input data to learn robust representations. Oppositely, contrastive approaches maximizes the mutual information between different views of the graph to learn representations. These contrasting methodologies offer distinct strategies for self-supervised learning on graph data.

This paper is a comparative analysis between two prominent paradigms in graph SSL: GraphMAE and MaskGAE. The former, introduced in [7], represents a novel model that combines SSL with a generative AE framework tailored specifically for graphs. In contrast, the latter leverages a contrastive SSL approach applied within a VAE framework as proposed by [17].

The primary objective of our study is to evaluate the strengths and weaknesses of these two approaches, taking into account factors such as structural robustness and real-world performance across diverse datasets. We aim to highlight the advantages and limitations of both GraphMAE and MaskGAE, providing valuable insights into their optimal use cases and applications. Ultimately, our findings seek to inform future research directions and practical implementations of graph SSL methods within the field of GNNs.

In the subsequent section, we begin with the theoretical foundations of graph representations and GNNs, followed by an exploration of the historical and motivational aspects of SSL, including an explanation of pretext and downstream tasks. We then shift focus to graph SSL, defining its two subtypes: generative and contrastive SSL. Moving into the approach section, we conduct a structural comparison of these subtypes and describe the baseline implementations of two prominent SSL methods, GraphMAE and MaskGAE. We proceed with a performance comparison between these methods, analyzing their strengths and weaknesses. Finally, we conclude with a discussion of our findings and their implications.

- *Maria Carmen Jica is a Data Science and Systems Complexity Master student at the University of Groningen, E-mail: m.c.jica@student.rug.nl.*
- *Chan Chan Tran is a Computing Science: Science, Business and Policy Master student at the University of Groningen, E-mail: h.c.c.tran@student.rug.nl.*

2 Literature Review

This section reviews existing research related to graph SSL and the use of GNN.

2.1 Theoretical Background

Graph Representations Real-world data exhibiting matrix-like relationships between objects often find their apt representation in the form of graphs [2]. These intricate structures, comprising vertices (nodes) and edges, manifest as complex networks that are versatile across numerous domains. From social networks to image pixel networks, from word networks in text to transportation networks, and from molecular structures to knowledge graphs [14, 20, 30, 8, 11, 15], graphs serve as a fundamental framework for data representation. When it comes to computational analysis, graphs can be summarized in one of three primary ways: through adjacency matrices, adjacency lists, or edge lists. The selection among these methods is contingent upon considerations such as size, density, memory constraints, and the specific operations to be performed [17].

Overview of Graph Neural Networks GNN belongs to a Deep Learning model and inherits neuron layers akin to traditional neural networks, but operating entirely on graph inputs. In each layer, GNN aggregates neighboring nodes and edges of each node, distilling them into an embedding, which guides the network's predictions. Following this, GNN updates the graph following a suitable loss function. Graph convolutional network (GCN), and graph attention network (GAT), and graph recurrent network (GRN) are some of the examples developed in this manner [13, 24, 21]. In particular, GCN has two main subtypes: spectral-based and spatial-based. Spectral-based leverages spectral graph theory to operate directly in the graph's Fourier domain, while spatial-based operates on a node's local neighborhood, focusing on the node's connectivity pattern [28].

2.2 Self-Supervised Learning

The innovation from supervised to SSL formed the cornerstone to the success of large machine learning models utilized in many domains, namely, Computer Vision (CV) and Natural Language Process-

ing (NLP). SSL, dubbed the "dark matter of intelligence" [1], was inspired by W.L.Taylor's cloze test, in which a portion of text is masked and the goal is to fill in the blank [23]. In modern days, the idea is adapted and generalized. The Turing award winner, Yann LeCun, described SSL as the process of predicting parts of its input from the observable part [1]. The cloze test is still widely diffused for NLP, such is the occlusion test for CV [31].

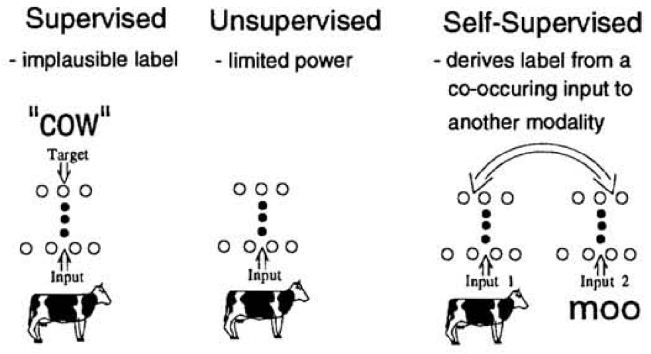


Figure 1. Cartoon-style representation of supervised, unsupervised and self-supervised learning [18]

Figure 1 depicts the there main learning paradigms in Machine Learning (ML) model training: supervised, unsupervised and self-supervised. The first leverages inherent labels in the dataset, in contrast to the second, which learns in the absence of labels. Lastly, SSL auto-generates pseudo-labels based on the information extracted from the large unlabeled data. Whilst unsupervised learning concentrates on detecting specific patterns, SSL is fixated on recreating either the entire graph structure or specific parts of the graphs through pretext tasks (e.g., graph reconstruction).

Pretext Tasks and Downstream Tasks Pretext tasks are auxiliary tasks designed to guide the model in acquiring knowledge during the training, consequently priming it for downstream tasks, i.e. real world applications or the ultimate end-goal. It is possible be quite creative and liberal with pretext task, however, there are typically two flavors: recovering the hidden data or its properties. Downstream tasks in graph-based machine learning encompass a diverse range, including node, link, and graph classification, clustering, and prediction. These tasks find applications in various real-world scenarios such as learning molecular fingerprints, predicting protein-protein interactions, and disease predictions in general practice [4, 9, 3]. Different configurations of self-supervised learning (SSL) tasks tailored for graphs yield varying performance outcomes in these downstream tasks. Jin et al. introduced AutoSSL, a framework designed to streamline the selection of multiple pretext tasks, optimizing their alignment with specific downstream objectives [10].

2.3 Graph Self-Supervised Learning Types

We will mainly highlight generative and contrastive learning since they are the types associated with the networks we account for in this paper. Despite that, it is important to acknowledge the presence of other subgenres categorized in previous literature: generative, contrastive and predictive in [27]; or generation, constrastive, auxiliary-property and hybrid-based in [19]

2.3.1 Graph Generative Learning

Graph Generative Learning (GGL) focuses on extracting meaningful information embeddings from the topological structure and the node features of the input graph with a reconstruction pretext task. Figure 2 presents the most conventional architecture, a typical AE model composed of an encoder and a decoder. The first component embeds the graph data into a latent space, then the decoder attempts to obtain the original structure.

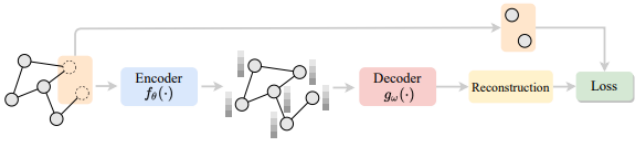


Figure 2. Generative learning [27]

2.3.2 Graph Contrastive Learning

Graph Contrastive Learning (GCL) is based on the intuition that similar data points would have close embedding and vice versa. Subsequently, the aim is to maximize the Mutual Information (MI) and minimize dissimilarities between multiple views from the same augmented instance (node, subgraph or graph), as summed up in Figure 3. A prime example is Deep Graph Infomax (DGI) by Veličković et al., which employs negative sampling to discriminate between positive and negative pairs of data samples [25]. Naturally, contrastive learning ties neatly with high graph discriminality or homophily, the principle that "like attracts like", as termed by Jin et al.. The underlying semantics of node u and v are more likely to be the same if they are connected by an edge. They further exploited this assumption and asserted that their AutoSSL is an improvement compared to DGI.

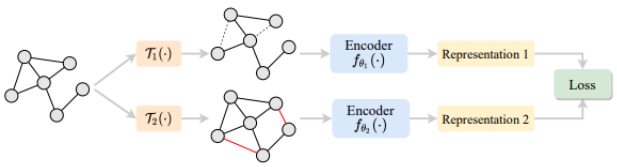


Figure 3. Contrastive learning [27]

2.3.3 Masking Strategy

There are several methods to augment input for GCL, namely, encompassing edge perturbation, diffusion, node dropping, random walk sampling, node attribute masking, and subgraph sampling. Simply put, these involves addition or removal of one or several elements of a graph.

Masked graph modelling (MGM) is a straightforward yet effective strategy in graph SSL. It was inspired from the success of masked image modeling (MIM) in CV [6] and masked language modeling (MLM) in NLP [22]. It refers to concealing or hiding a portions of the input data during training, which can encourage the model to be resilient and learn more generalized representations.

3 Structural Comparison

In the following section, we inspect and analyze the two self-supervised learning paradigms: GraphMAE and MaskGAE. We proceed by isolating specific design components and contrasting their respective approaches.

3.1 Baseline Implementation

GraphMAE is based on the generative self-supervised learning paradigm, which was, until recent studies, known to suffer limitations due to a number of issues hindering its progress: over-emphasizing proximity information, feature reconstruction is often affected by corruption, unsuitable error measure that leads to instability, scarce expressivity of decoder architectures. Therefore, GraphMAE employs an autoencoder as underlying implementation idea. The novelty arises from mitigating these limitations through a series of strategies. Firstly, the authors use the cosine error as an error criterion, thus aiming to

[1] https://ai.meta.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

balance the context of largely varying feature vector magnitudes. Secondly, a masking procedure is implemented on the node attributes and graph structure level. Thirdly, they facilitate a transition from the previously used decoder architecture, Multilayer Perceptron (MLP), to the more expressive graph neural nets (GNNs).

MaskGAE is situated on the other spectrum of self-supervised learning, being based on the contrastive paradigm. Recent literature is heavily oriented towards this approach, confirming its position as the lead technique in the field currently. Despite its popularity, contrastive methods are also facing temporary stagnation. Its strong points in the form of data augumentation and highly specialised pretext tasks represent a point of blockage. The authors propose to overcome these obstacles by using MGM as a pretext task.

3.2 Framework

GraphMAE builds upon the Graph AutoEncoder (GAE) framework, where a graph is represented as $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{X})$, with $\mathcal{V}$ denoting the node set, $\mathcal{A}$ as the adjacency matrix, and $\mathcal{X}$ as the input node feature matrix. The graph encoder $\mathtt{f}_\mathtt{E}$ and decoder $\mathtt{f}_\mathtt{D}$ aim to reconstruct the input graph by encoding it into a latent code $\mathcal{H}$ and then decoding it back into $\mathcal{G}'$. Notably, various GNN options are available for $\mathtt{f}_\mathtt{E}$ and $\mathtt{f}_\mathtt{D}$, with specific choices depending on the graph learning tasks.

In GraphMAE, the focus lies on feature reconstruction as the primary training objective. This is achieved by masking a subset of node features with a special token [MASK], ensuring that the model learns to reconstruct the masked features from the partially observed node signals $\widetilde{\mathcal{X}}$. Additionally, a re-masking decoding technique is proposed, wherein the latent code $\mathcal{H}$ is processed for decoding by replacing masked node indices with another mask token [DMASK]. The choice of encoder and decoder architectures depends on the specific task requirements, with GAT and Graph Isomorphism Network (GIN) [29] often preferred for node and graph classifications, respectively.

To measure the reconstruction quality, GraphMAE employs the Scaled Cosine Error (SCE) loss, which penalizes the deviation between the original features $\mathcal{X}$ and the reconstructed output $\mathcal{Z} = \mathtt{f}_\mathtt{D}(\mathcal{A}, \widetilde{\mathcal{H}})$. This loss function incorporates a scaling factor $\gamma\mathcal{A}$ to gradually adjust the sample weights based on the reconstruction error.

In MaskGAE, MGM aims to learn a graph encoder $\mathtt{f}_\theta$ that maps the input graph $\mathcal{G}$ to low-dimensional latent representations $\mathcal{Z}$. Within the GAE framework, MaskGAE leverages naturally occurring pairs of similar and dissimilar nodes in a graph as self-supervised signals. By adopting the information-maximization (infomax) viewpoint of contrastive learning, MaskGAE maximizes the MI between k-hop subgraphs of adjacent nodes. This is achieved by minimizing a binary cross-entropy loss function, where the decoder reconstructs positive and negative node pairs sampled from the graph.

3.3 Masking strategy

Masked autoencoding currently represents one of the most promising innovations in the field of self-supervised learning. The learners investigated in this paper have identified this potential space for progress in the context of graph modeling. Therefore, both paradigms employ the strategy of incorporating masked elements/features, but from different perspectives.

On the one hand, GraphMAE proposes applying the masking procedure on the graph data level directly, by using the obstruction of certain structures of the graph and of the node attributes. On a high level, the maskage approach is used as the learning strategy for the purpose of self-supervised learning. This system, incorporated in both the encoder and the decoder architectures, can be observed in figure 4. Initially, GraphMAE obscures certain input node features before the training process commences. The altered graph is then fed to the encoder, which is a general GNN encoder. The subsequent step is to re-mask the part of the resulting code that has previously been marked, with a separate token before being inputted into the GNN decoder. The concerning features are, lastly, reconstructed. This process aims to extract information from the reconstructed features, which can be considered the critical element of the approach.

On the other hand, MaskGAE benefits from the organic compatibility between contrastive learning and masked modeling. This natural rapport is created by several factors that allow the two techniques to harmoniously complement each other towards a common goal. First of all, the complementarity of their respective objectives can be easily deduced. While masking focuses on extracting relevant features by obstructing irrelevant ones, contrastive methods shift the focal point toward learning discriminative representations. Secondly, triviality in a large overlapping subgraph can be avoided by hiding edge information from the original paired subgraphs. Following this train of thought, MaskGAE incorporates masking strategies in the used pretext tasks. Two strategies have been utilised:

- **Edge-wise masking.** A trivial and frequent masking technique that operates by randomly sampling a subset of edges that follow a particular data distribution.
- **Path-wise masking.** A novel approach of masking, where the sample unit is represented by a path, therefore obstructing any connections or elements linked to the unit. The result is a learner that requires capturing deeper features about the data, as trivial information is no longer available.

3.4 Overcoming Limitations in Graph Learning

The designs of both MaskGAE and GraphMAE stem from the defining need of overcoming the limitations imposed by the respective paradigms. As such, each design choice comes as a response to an recognized issue.

A thorough understanding is needed to enable the selection of a viable data perturbation method. For instance, random node dropping is beneficial for training in social networks, but is detrimental in molecular graphs. Furthermore, GCL suffers from unstable learning in early stages due to its sensitivity to hyperparameters and network architecture. In case of a low feature discriminalty, it is unclear which loss function to use as the incorrect choice will lead GCL to overfitting, vanishing gradient and divergence in the worst-case, consequently slow convergence and poor generalization. Finally, since GCL relies on the synergy of its elaborate components, it becomes imperative to conduct many experiments to discover the most optimal settings and increasing implementation complexity.

Aside from these disadvantages of contrastive learning, Hou et al. defend GGL's position in graph representation learning through Self-Supervised Masked Graph Autoencoder (GraphMAE) as a proof of concept. The authors identified the four common downsides of GGL that hindered its widespread adoption. Firstly, GGL may overemphasize proximity information between neighboring nodes, useful for link prediction and node clustering, at the expense of higher-level structural details relevant for classification. Secondly, they are susceptible to noise as they do not incorporate corruption, decreasing the robustness of feature reconstruction in the presence of unseen data. Another issue is connected to the sensibility and instability of mean square error. This loss function suffers from the elevated variability in magnitude of feature values due to the inherent high-dimensionality characteristic of graph data. As a result, the model struggles to converge and causes the training to collapse. The fourth problem adheres to the limitation of the basic decoder function, multilayered perception, which is incapable of capturing the incomprehensive semantic, risen when graph data is translated into a vector.

Figure 4 explains the steps taken in GraphMAE to mitigate these challenges, with the inclusion of three innovative aspects: node features reconstruction with masks, GNN as decoder on re-masked hidden space, and SCE. The researchers investigated the comparative effectiveness of node features versus edges as a reconstruction target and found that the former outperforms the latter in downstream classification tasks. To tackle the second challenge, they implemented a denoising strategy by applying double masking just before the encoder

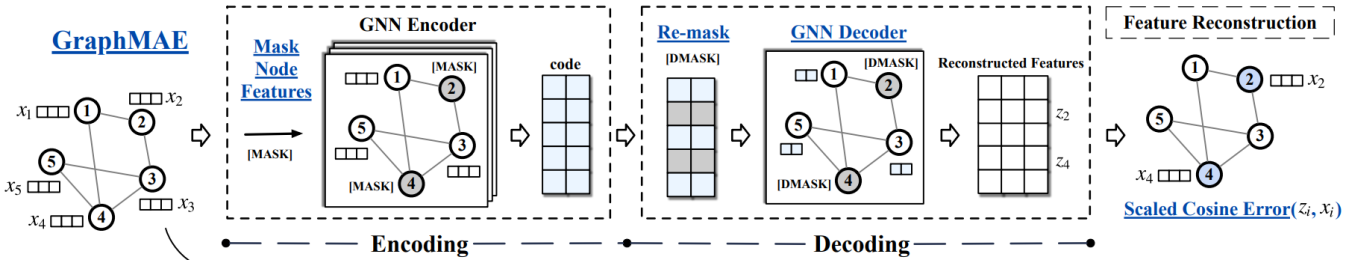


Figure 4. The architecture of GraphMAE. [7]

and the decoder. Additionally, they identified the SCE as a more robust metric for assessing the pretext task's quality. To enhance the model's capability to capture low-informative features, they replaced the simplistic decoder with a GNN.

There are also several obstacles posed on the current GCL architectures: sensitivity of MI, imbalances between local and global information, restriction of edge masking and scalability. Li et al. demonstrate that by integrating GCL with MGM, we can enhance the mutual information maximization by reducing overlap between subgraph views.

MaskGAE is tailored by a asymmetric design, where a decoder is followed by two decoders. The encoder is modified from the traditional graph AE to a flexible GNN. The paper decided on a GCN, however, it is adaptable to other networks such as GraphSAGE or GAT as demonstrated in the ablation study. MaskGAE utilizes a structure and a degree decoder. The former is a MLP and a sigmoid activation function that aggregates pairwise node to unveil the masked link representation. The latter is auxiliary whose goal is to adjust the second aforementioned issue. Li et al. employed a dual-loss function, reconstruction and regression loss, to regularize the encoder. The first measures the proficiency of the edge reconstruction while the other quantifies the closeness between the approximated and the original node degrees. The representation of the aforementioned architecture can be observed in Figure 5.

3.5 Performance Comparison

In the following section, we will be describing a qualitative comparison between MaskGAE, GraphMAE and the original benchmark for graph components classification, GAE. The types of evaluation criteria that will be used are Area Under Curve and average accuracy for link prediction and accuracy percentage for node classification.

Dataset The paper [17] features a series of experiments where both learning techniques are being compared. The experiments were conducted on 8 benchmark datasets, whose statistics and configurations are detailed in table 1. The paper proceeds to mention that for the link prediction task they have used a random split of 80%5%10% of edges for training/validation/testing, respectively, for all datasets, with the exception of Collab, for which they have used public splits. For the node classification task, a $1:1:8$ random split strategy for train/validation/test for Photo and Computer, while the remaining collections used public splits.

Dataset	#Nodes	#Edges	#Features	#Classes	Density
Cora	2,708	10,556	1,433	7	0.144%
CiteSeer	3,327	9,104	3,703	6	0.082%
Pubmed	19,717	88,648	500	3	0.023%
Photo	7,650	238,162	745	8	0.407%
Computer	13,752	491,722	767	10	0.260%
arXiv	16,9343	2,315,598	128	40	0.008%
MAG	736,389	10,792,672	128	349	0.002%
Collab	235,868	1,285,465	128	-	0.002%

Table 1. Dataset statistics. [17]

3.5.1 Performance Comparison on Link Prediction

The data includes the performance of MaskGAE, GraphMAE, and, additionally, of the GAE predecessor, in order to have a comparative benchmark that would highlight the improvement, if any, over the original framework. Furthermore, we note the existence of 2 types of results for MaskGAE, the difference being made by different masking techniques with alternative objectives: path-wise and edge-wise strategies. This convention is kept across all of the presented results.

As GraphMAE was not originally created with the clearly defined goal of performing link prediction in mind, an additional component, an MLP decoder, has been introduced and trained. Moreover, the used evaluation metrics are composed of Area Under Curve (AUC) and average precision (AP). The results can be observed in table 2 and 3.

	Cora	CiteSeer	Pubmed
GAE	91.09 ± 0.01	90.52 ± 0.04	96.40 ± 0.01
GraphMAE	94.88 ± 0.23	94.32 ± 0.40	96.24 ± 0.36
MaskGAE_{edge}	96.42 ± 0.17	98.02 ± 0.22	98.75 ± 0.04
MaskGAE_{path}	96.45 ± 0.18	97.87 ± 0.22	98.84 ± 0.04

Table 2. Comparison of link prediction (Area Under Curve) over several benchmarks. [17]

	Cora	CiteSeer	Pubmed
GAE	92.83 ± 0.03	91.68 ± 0.05	96.50 ± 0.02
GraphMAE	93.52 ± 0.51	93.54 ± 0.22	95.47 ± 0.41
MaskGAE_{edge}	95.91 ± 0.25	98.18 ± 0.21	98.66 ± 0.06
MaskGAE_{path}	95.95 ± 0.21	98.09 ± 0.17	98.78 ± 0.05

Table 3. Comparison of link prediction (average precision) over several benchmarks. [17]

We can observe that both GraphMAE and MaskGAE have a significant increase in performance compared to the original approach, GAE. MaskGAE has outperformed the other 2 on both criterion categories, with the highest performance improvement being a remarkable 5%. GraphMAE has consistently achieved impressive scores. It has almost exclusively placed second, with a single exception.

Furthermore, Li et al. have also conducted the same experiment, under the same setting and configurations, for several more graph learners: VGAE [12], ARVGA [16], SAGE [5], SEAL [33], MaskGAE [26]. Even with the incremented set of candidates, MaskGAE still scored the highest in all of the benchmark datasets for both evaluation metrics. GraphMAE has placed 3^{rd} overall, marginally being outranked only by MaskGAE. We can observe the trend that the chosen frameworks for this project are consistently delivering promising results, outcome which we associate with the integration of the MGM paradigm in the architectures.

3.5.2 Performance Comparison on Node Classification

For the purpose of the node classification task, the same configuration has been kept, with the specification that the model for MaskGAE

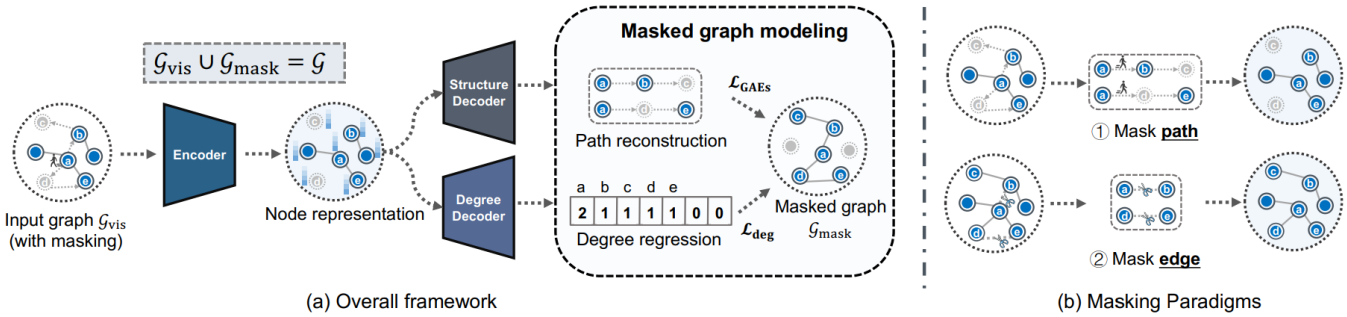


Figure 5. The architecture of MaskGAE. [17]

has been trained on a partially corrupted graph. For each of the contrastive and generative methods, we included the results of the best performers, aside from MaskGAE and GraphMAE, in the used collection: ARVGA [16] and CCA-SSG [32]. These tools were selected out of a pool of 17 candidates to have a further source of comparative information, now in a broader context of graph learners. The outcome values can be observed in table 4.

We can easily observe that MaskGAE and GraphMAE achieve the leading performance in their respective categories of graph learning techniques. MaskGAE once again attains the highest accuracy result, not only among contrastive candidates but also overall. Furthermore, GraphMAE is arguably the best application of the generative paradigm, obtaining an enhanced accuracy by approximately 7% for the CiteSeer benchmark compared to its next competitor. In addition, GraphMAE also accomplished the remarkable objective of outperforming the majority of contrastive methods, which until recently have been proven to be the superior alternatives.

4 Discussion & Findings

One crucial aspect of this study is the parallel between contrastive and generative learning methods, two fundamentally different paradigms attempting to solve equivalent problems. Through the lens of masked modeling, we scrutinized frameworks associated with both paradigms, acknowledging the nuanced interplay between generative and contrastive approaches. While it is true that MaskGAE embodies a mixture of generative and contrastive learning through its employment of masking techniques, it distinctively operates within the realm of informative theory, setting it apart from traditional Euclidean-based approaches like GraphMAE.

The success of MaskGAE, ranking top regardless of dataset or evaluation metric, underscores its efficacy and versatility across various configurations of contrastive learning methods. Additionally, the utilization of decoders for multiple auxiliary tasks in MaskGAE opens a promising avenue for further investigation, suggesting the potential for assigning multiple decoders as domain-specific experts. However, generative methods have also shown improved performance and is significant in its own space. In fact, GraphMAE consistently placed among the optimal-performing learners, winning over the majority of contrastive or generative alternatives.

We attribute part of the success of GraphMAE and MaskGAE to the integration of the MGM paradigm in their respective architectures. MGM is a good fit for self-supervised learning, especially in the context of graph modeling, stems from its ability to mitigate redundancy and seamlessly align with SSL objectives. There are great benefits in training a model on a partially visible graph, as the transition towards the edge prediction task is made more seamless. Therefore, contrastive and generative methods could benefit from additions of masked strategies, regardless of whether the integration is done under the form of pretext task modeling, training phase obscuring techniques, choice of element corruption, etc. Further exploration of masked graph modeling is deemed essential, given its promising trajectory for the field.

Choosing between GraphMAE and MaskGAE depends on the specific needs of the task at hand and the characteristics of the data being analyzed. GraphMAE is simpler to implement while offering a strong solution for scenarios where generative learning is favored, such as when generating new graph instances or reconstructing graph structures for downstream tasks like classification. Its innovative strategies, including cosine error optimization and the use of masking procedures, enhance its ability to capture complex graph representations. However, GraphMAE may not perform optimally in tasks requiring discriminative representations, as its focus leans more toward generative modeling.

Contrarily, MaskGAE shines in scenarios where contrastive learning is preferred, particularly in tasks necessitating the identification of relevant features amidst noisy or irrelevant data. Its flexibility in selecting masking strategies allows adaptation to various datasets and objectives, making it versatile for different applications. Nonetheless, MaskGAE relies on complicated and elaborate designs and requires more computational power and memory resources to complete a high number of iterations from multiple contrasting graph views. Moreover, it may not be as effective in tasks prioritizing the preservation of structural details or feature reconstruction, as its primary focus lies in contrastive learning. In summary, there are trade-offs between the two approaches; factors like the nature of the data, desired outcomes and resource constrains can aid in selecting the most suitable method for a given task.

Additionally, the success of MGM in these initial integration attempts should encourage future experts to explore the realm of potential that it holds. In particular, domain-specific applications could represent an essential step forward. Real-world, isolated functions such as social networks, biological networks, or knowledge graphs encapsulate highly specialized patterns that could be naturally captured by a highly specialized model. Therefore, specialized models uncover new opportunities for advancing the field and addressing real-world challenges.

5 Conclusion

In this paper, we have conducted a comprehensive investigation of two distinctive self-supervised learning paradigms for graph representation. The study applies a comparative analysis, discussing the primary components and critical design choices for each architecture. We utilised a specialised outlook that concentrates on the integration of MGM principles and the affiliated outcomes. We discovered that MGM benefits the learning process greatly due to eliminated redundancy, hereby encouraging the extraction of relevant features from the data. The benefits of MGM can be feasibly observed and argued for, as both GraphMAE and MaskGAE have achieved good computational scaling and downstream performance across all benchmarks.

Acknowledgements

The authors wish to thank Huy Truong for supporting this paper.
We also wish to thank the authors of the [7] and [17] papers, without whose contributions and meaningful research, this paper would not be possible.

	Cora	CiteSeer	Pubmed	Photo	Computer	arXiv	MAG
GAE	74.90 ± 0.40	65.60 ± 0.50	74.20 ± 0.30	91.00 ± 0.10	85.10 ± 0.40	63.60 ± 0.50	27.10 ± 0.30
ARVGA	79.50 ± 1.01	66.03 ± 0.65	81.51 ± 1.00	91.51 ± 0.09	86.02 ± 0.11	67.43 ± 0.08	28.32 ± 0.18
GraphMAE	84.20 ± 0.40	73.40 ± 0.40	81.10 ± 0.40	93.23 ± 0.13	89.51 ± 0.08	71.75 ± 0.17	32.25 ± 0.37
CCA-SSG	83.59 ± 0.73	73.36 ± 0.72	80.81 ± 0.38	93.14 ± 0.14	88.74 ± 0.28	69.22 ± 0.22	27.57 ± 0.41
MaskGAE$_{edge}$	83.77 ± 0.33	72.94 ± 0.20	82.69 ± 0.31	93.30 ± 0.04	89.44 ± 0.11	70.97 ± 0.29	32.75 ± 0.43
MaskGAE$_{path}$	84.30 ± 0.39	73.80 ± 0.81	83.58 ± 0.45	93.31 ± 0.13	89.54 ± 0.06	71.16 ± 0.33	32.79 ± 0.32

Table 4. Comparison of node classification accuracy (percentage) over several benchmarks.[17]

REFERENCES

[1] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum. A cookbook of self-supervised learning, 2023.

[2] G. Blelloch and M. Reid-Miller. Chapter 9: Graphs: Definition, applications, representation, 2024. URL https://www.cs.cmu.edu/afs/cs/academic/class/15210-s14/www/lectures/graphs.pdf. Lecture notes.

[3] S. Chi, Y. Wang, Y. Zhang, W. Zhu, and J. Li. Graph neural network based multi-label hierarchical classification for disease predictions in general practice. *Studies in health technology and informatics*, 310:725–729, 01 2024. doi: 10.3233/SHTI231060.

[4] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints, 2015.

[5] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

[6] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009, June 2022.

[7] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang. Graphmae: Self-supervised masked graph autoencoders, 2022.

[8] D. Jana, S. Malama, S. Narasimhan, and E. Taciroglu. Edge ranking of graphs in transportation networks using a graph neural network (gnn), 2023.

[9] K. Jha, S. Saha, and H. Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12:8360, 05 2022. doi: 10.1038/s41598-022-12201-9.

[10] W. Jin, X. Liu, X. Zhao, Y. Ma, N. Shah, and J. Tang. Automated self-supervised learning for graphs, 2022.

[11] Y. Kim, Y. Jeong, J. Kim, E. K. Lee, W. J. Kim, and I. S. Choi. Molnet: A chemically intuitive graph neural network for prediction of molecular properties, 2022.

[12] T. N. Kipf and M. Welling. Variational graph auto-encoders, 2016.

[13] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks, 2017.

[14] J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining Social-Network Graphs*, page 325–383. Cambridge University Press, 2014.

[15] J. Li, H. Shomer, J. Ding, Y. Wang, Y. Ma, N. Shah, J. Tang, and D. Yin. Are message passing neural networks really helpful for knowledge graph completion?, 2023.

[16] J. Li, S. Tian, R. Wu, L. Zhu, W. Zhao, C. Meng, L. Chen, Z. Zheng, and H. Yin. Less can be more: Unsupervised graph pruning for large-scale dynamic graphs, 2023.

[17] J. Li, R. Wu, W. Sun, L. Chen, S. Tian, L. Zhu, C. Meng, Z. Zheng, and W. Wang. What's behind the mask: Understanding masked graph modeling for graph autoencoders, 2023.

[18] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2021. ISSN 2326-3865. doi: 10.1109/tkde.2021.3090866. URL http://dx.doi.org/10.1109/TKDE.2021.3090866.

[19] Y. Liu, M. Jin, S. Pan, C. Zhou, Y. Zheng, F. Xia, and P. Yu. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2022. ISSN 2326-3865. doi: 10.1109/tkde.2022.3172903. URL http://dx.doi.org/10.1109/TKDE.2022.3172903.

[20] A. Newell and J. Deng. Pixels to graphs by associative embedding, 2018.

[21] L. Ruiz, F. Gama, and A. Ribeiro. Gated graph recurrent neural networks. *IEEE Transactions on Signal Processing*, 68:6303–6318, 2020. ISSN 1941-0476. doi: 10.1109/tsp.2020.3033962. URL http://dx.doi.org/10.1109/TSP.2020.3033962.

[22] K. Sinha, R. Jia, D. Hupkes, J. Pineau, A. Williams, and D. Kiela. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021. doi: 10.18653/v1/2021.emnlp-main.230.

[23] W. L. Taylor. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, Sep 1953. doi: 10.1177/107769905303000401.

[24] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks, 2018.

[25] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax, 2018.

[26] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang. Mgae: Marginalized graph autoencoder for graph clustering. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 889–898, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349185. doi: 10.1145/3132847.3132967. URL https://doi.org/10.1145/3132847.3132967.

[27] L. Wu, H. Lin, Z. Gao, C. Tan, and S. Z. Li. Self-supervised on graphs: Contrastive, generative, or predictive. *CoRR*, abs/2105.07342, 2021. URL https://arxiv.org/abs/2105.07342.

[28] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, Jan. 2021. ISSN 2162-2388. doi: 10.1109/tnnls.2020.2978386. URL http://dx.doi.org/10.1109/TNNLS.2020.2978386.

[29] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks?, 2019.

[30] S. Yang, Y. Liu, Y. Zhang, and J. Zhu. A word-concept heterogeneous graph convolutional network for short text classification. 55(1):735–750, jun 2022. ISSN 1370-4621. doi: 10.1007/s11063-022-10906-6. URL https://doi.org/10.1007/s11063-022-10906-6.

[31] X. Zhan, X. Pan, B. Dai, Z. Liu, D. Lin, and C. C. Loy. Self-supervised scene de-occlusion, 2020.

[32] H. Zhang, Q. Wu, J. Yan, D. Wipf, and P. S. Yu. From canonical correlation analysis to self-supervised graph neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 76–89. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/00ac8ed3b4327bdd4ebbebcb2ba10a00-Paper.pdf.

[33] M. Zhang, P. Li, Y. Xia, K. Wang, and L. Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 9061–9073. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/4be49c79f233b4f4070794825c323733-Paper.pdf.

A Comparison Between Several Load Balancing Methods in Data Center Networking

Joe Maaßen, Rutger Wuijster

Abstract— In the modern work of high-performance datacenters, a pressing issue arises more and more. While the servers within the datacenters seem to be able to produce, or swallow, large amounts of data, the networking between the servers is a large bottleneck. Making use of the networks between each server within a datacenter is not just a matter of picking the shorted physical path between two machines, but rather becomes a complex decision task of selecting less congested paths, and keeping the order of data being sent. In this paper, we will discuss four different methods of balancing the networking within datacenters: CONGA; Prest; Clove; W-ECMP.

Index Terms—Datacenter networking, Complex path decision making, CONGA, Presto, Clove, W-ECMP.

1 Introduction

Networking within datacenters is generally considered to be bursty. This means that each application running on servers within the datacenter does not require a constant and low bandwidth, but rather uses very high bandwidth for short periods of time, a burst of data. Examples of this would be high performance computing applications accessing data stored within a database on a different server, completing heavy calculations on this, and finally returning the results to another site in the datacenter. In the past papers on VL2 [7] and Portland [1] try to address these issues using Clos topologies, Equal Cost MultiPath (ECMP) load balancing, and the decoupling of endpoint addresses from their location. Later generations of overlay technologies follow the same design principles, with VXLAN [10] and NVGRE [5]being notable examples.

Unfortunately, several studies have shown that ECMP performs sub-optimally [8, 11] when a small number of large flows are happening concurrently over the networks, due to hash collisions. Another case where ECMPs performance is worse is within asymmetric systems due to the local only traffic splitting, while it is possible that there is traffic congestion downstream, making ECMP worse in cases of link failures causing asymmetric topologies, something that happens frequently in datacenters [6].

To combat these shortcomings of network balancing methods that make use of ECMP load balancing, a paper by Alizadeh *et al.* proposed CONGA [2]. CONGA is a network-based distributed congestion-aware load balancing mechanism for datacenters. This means that CONGA is based on networking, and uses congestion metrics to make load balancing decisions. In later papers CONGA is used as a base to test against [9], showing that CONGA is generally a better load balancer compared to ECMP and setting a new benchmark.

When switching over to a different load-balancing algorithm like CONGA it is important to take into account the ease of deployment. In the case of CONGA, like will be discussed in further sections, requires the usage of custom ASICs [2]. This means that to be able to use CONGA, a datacenter has to replace physical network equipment, which can be costly due to the fact that it requires physical labour with some switches weighing close to 6kg [4], as well as longer down times compared to over the air updates, as it requires maintenance workers to disconnect all connections into a switch and reconnecting all these connections once the new switch is installed. Clove [9] makes it a lot easier to deploy as it is a hypervisor-based load balancer. This means that the algorithms used for Clove are all based in software and thus can easily be deployed to all kinds of systems.

W-ECMP (Weighted-ECMP) seeks to be a protocol and hardware agnostic solution that distributes incoming network traffic across all available paths. It does this by measuring the current level of congestion and creating a probability for each path to be chosen.

Next, in section 2 we will be summarizing the four methods mentioned above. In section 3 we will be discussing the advantages and disadvantages of each method and make some comparisons between the methods. Finally, we will draw a conclusion, showing which method is better suitable in which situation in section 4.

- *Joe Maaßen is a master student at the university of Groningen, E-mail: j.maassen.1@student.rug.nl*
- *Rutger Wuijster is a master student at the university of Groningen, E-mail: rutgerwuijster@hotmail.com.*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org .

2 Summaries

To get a clear picture of all for load balancing methods that were introduced in the introduction, a summary of all four methods mentioned above will be given.

2.1 Flowlet

Multiple of the previously defined methods, and many others make use of something called a 'flowlet'. Within networking a stream of data from one source computer to one destination computer can also be named a flow. When this flow is divided up into chunks each smaller flow is called a flowlet. Flows are usually cut after a stream has a large gap between two data packets, signifying the end of a flowlet, followed by the beginning of a new flowlet. This concept of flows and flowlets is used in many different networking systems and thus helpful to grasp.

2.2 CONGA

CONGA keeps track of a local congestion variable and piggybacks this from switch nodes to leaf nodes. Then using the congestion metrics, that are stored in a table on a per destination per link basis on each leaf node. Each time a packet is sent to a destination a flowlet is either created or followed. If there is a current active flowlet in the same direction as the packet, then the packet will use the associated port on the source switch. When there is not a valid flowlet from the source to the destination nodes a new flowlet is created. In this case the congestion metric table is checked and a flowlet is chosen to minimize the maximum congestion metric.

The local congestion variable, or Discount Rate Estimator (DRE) as it is called in the paper, is calculated by using a single register storing the congestion variable. Every time a new packet arrives, the congestion variable is increased by the size of the packet in bytes. Then a timer is set to go off periodically, decreasing the local congestion variable multiplicatively. Following the equation $X_1 = X_0 * (1 - \alpha)$ where X_0 is the old congestion variable, X_1 is the new congestion variable,

and α is the factor by which each period the congestion factor must decrease. This way it is possible for a switch node to keep track of its local congestion

CONGA piggybacks the congestion metrics between leaf nodes by encapsulating a packet being sent over the network using the VXLAN [10] encapsulation format. In this encapsulation 4 separate data are stored:

- LBTag: 4 bits of data that partially identify the path the packet is taking
- CE: 3 bits used by all switches to convey the congestion along the path the packet is taking
- FB_LBTag: 4 bits used for piggybacking the path back from a destination to a source node
- FB_Metric: 3 bits used for piggybacking the congestion along the path FB_LBTag from a destination to a source node

Using the data within the encapsulation it is possible for CONGA to send link congestion information between leaf nodes. When a packet is sent, the LBTag section is set to the link port the packet is being sent from in the source leaf node. When traveling through a spine node, the CE bit is compared to the local congestion variable and the higher one is set into the CE bit. When the packet arrives at the destination leaf the maximum congestion along the taken path is stored in CE and is remembered on a per source leaf per port basis. Finally, when a packet is sent back from the destination leaf to the source leaf the FB_LBTag value is set to be the LBTag value, and the FB_Metric value is set to the CE tag. This might make it seem like only one of the two happens at any time, but both happen at the same time, making this an efficient manner of moving the metrics between each node.

2.3 Presto

The core approach of Presto is it to shift the process of managing the data flows into the soft edge. This enable Presto to operate independent of the underlying hardware. Presto will divide all incoming flowlets into multiple smaller flows that travel through the network equally distributed before being recombined at the destination.

The smaller flows are called flowcells and are defined to be a uniform size of 64 kB. This size was chosen as a compromise between increasing the granularity of the system and supporting high speed network applications. The final size was determined by matching it to the maximum segment size of the existing TCP segmentation offloading (TSO) standard since it is already integrated in existing systems.

Presto uses an approach for load balancing where it does not measure actual congestion inside the network and as a result will not react to it. It will only react when a network event impedes the normal functionality. The underlying network topology is analysed by a centralized controller that also sends that information to the vSwitches.

When deciding the path trough the network Presto uses a End-to-End-Multipathing solution. There it decides the whole path each flowcell will take before sending it to the final destination. As a result each flowcell of a flow can take a different path and arrive out of order.

The most challenging component of Presto is the reordering of the received packages. The paper discusses the issues that can appear in the process and explains how they have handled them.

2.4 Clove

Clove is a hypervisor based load-balancer meaning that Clove can be installed on hardware that is already present in the server, by uploading new frimware, or updating older Clove versions already present on the hardware. Clove will send out periodic packets that are used to detect paths that might exist between the source hypervisor and destination hypervisors. This periodic sending of packets is called Traceroute, where each packet is forced to take a random path from the source hypervisor to the destination hypervisor. This way it is possible for the source hypervisor to detect when there exist shorter paths to certain destination hypervisors, allowing for fast switching to less congested, or higher bandwidth paths. As an optimization, the Traceroute algorithm is only allowed to test the random path times to the destination hypervisors that destination hypervisor is currently getting sent traffic from the source hypervisor. This periodic sending of packets, also known as probing, is done with a specific encapsulation header that allows switches in between the source and destination hypervisor to continue using ECMP, so that this hardware does not need to be altered.

Clove uses the routing of flowlets, sections of a traffic flow with a time gap between them of specified or more time. The routing of flowlets is used so that there is a low chance of packets arriving at a destination out of order. This means that, while sending a flowlet, all packets will use the same path, even when another, faster path has been found by the Traceroute algorithm. Once the flowlet time gap specified amount of time has passed since the sending of the last packet and a new packet is sent, a new flowlet is created and all subsequent packets that belong to the new flowlet are sent over the previously discovered faster path. In the paper, three varieties of path selection techniques that distribute the flowlets are discussed, each subsequent technique being more sophisticated and achieving higher performance.

The first technique of path selection discussed is called Edge-Flowlet. This technique chooses a new path at random, not taking into account the congestion of the network. While this seems counter productive, there are two possible reasons for a flowlet gap to exist. The first reason is the source not having any data to actually send over the network. The second reason is the possibility of the old flowlet having been routed over a congested path, resulting in a slow return of the acknowledgement packets, meaning that the switching of paths for different flowlets can be considered a sign of congestion. In the second case there exists a possibility that the flowlet is part of a much larger flow sending large amounts of data over the network, colliding with a different equally large flow. In this case the random selection of different paths, while not optimal, is already better than ECMP as this would continue to send the large flow over the same path.

The second technique of path selection that is proposed is named Clove-ECN. This technique makes use of a built-in feature on modern networking hardware called Explicit Congestion Notification (ECN). ECN is used to indicate congestion to a source, and have the source throttle back to not overload the network. A source is able to indicate that it can react to ECN feedback, by setting the ECN Capable Transport (ECT) bit in the IP header of a packet. Then, along the path of the packet ECN capable switches will use the Congestion Experienced (CE) bits in the IP header when the packet enters a queue that is larger than a set value. Finally the destination sends this ECN information back to the source, that can then throttle back until the congestion is cleared.

Clove-ECN uses this ECN information to detect congestion along each path the flowlets take to the destination, and weighs each path accordingly. Then, when another flowlet is initiated, using the weights added to each path, using weighted round robin (WRR), the next path for a flowlet to take is chosen. Since each ECN information return changes the weights for the WRR algorithm, an optimization that can be made is not relaying each ECN information packet, but returning one every few round trip times. This way the source processor will spend less time recalculating the weights of each path, and will make the response to congestion less aggressive.

The final technique mentioned is Clove-INT. This technique makes use of an upcoming feature named In-band Network Telemetry (INT). At the time that the paper on Clove was written, 2017, it was assumed that INT would be a feature that would become available on newer networking hardware. This is the case as Cisco has documentation on INT going back to early 2019 [3]. INT enabled devices attach current information on link usages, compared to link congestion allowing Clove-INT to react to link usage and spread out flowlets over less used links, thus proactively rerouting flowlets to less used links rather than reacting to over-used and congested links.

2.5 W-ECMP

Weighted Equal-cost multi path (W-WCMP) is programmed directly into the data-plane of the network using the P4 language.

W-ECMP proposes a header that can be attached to any protocol to communicate the current congestion of the connection. The header contains four fields that each contain one byte of information.

As a flowlet progresses trough the network the header will be modified to update the contained information. The core information is the max_utilisation. It is measured at each port and translated into a weight. The lower the weight the more the port is used. When the flowlet passes through a connection that has a lower weight than what is currently stored inside the W-ECMP header it replaces it. The information about the current usage percentage is collected through a module that computes the transmission rate of each port.

In the paper each switch is assigned a tag id of zero or one and when the flowlet passes trough the switch it modifies a bit inside the tag_path_id field. The first switch modifies the right most and all following ones modify the bit to the left of the previous modification. The selected_path_id uses the same information but instead it is preset when the flowlet is sent out and read at each switch to determine the switch the package should be directed towards.

After the flowlet has arrived at its destination the data for the given path is collected. By combining the weights of all possible paths between two leaf switches a utilisation table is generated. The table contains the weights for each path and ranges of numbers based on the weights. When a new flowlet is send between the two leaf switches a random number is chosen that is contained in the total range. The flowlet will take the path the random number points to.

3 Comparison

Now that it is clear how each of the load balancing methods works, it is time to discuss the advantages and disadvantages of each method.

3.1 CONGA

Since CONGA makes use of congestion metrics it is possible to make decisions that take into account all congestion along a path between leaf nodes. Combined with the fact that these congestion metrics are piggybacked on the packets being sent over the network compared to using extra packets to send this information, CONGA uses a robust way of sharing the state of network congestion without adding extra congestion, and making use of this to make better decisions compared to ECMP making local decisions.

To implement CONGA the paper highlights the use of special ASICs. Using two different ASICs the researches implemented leaf ASICs with flowlet detection, congestion metric tables, local congestion metric algorithms, and leaf-to-leaf feedback mechanisms, and spine ASICs that implement the local congestion metric algorithm along with its associated way of marking the packets that pass the spine. Even tough both ASICs promise exceptional performance while using very little die area to be implemented, it is proprietary technology nonetheless. This means that, instead of updating networking equipment, to gain the better performance of CONGA it is necessary to replace entire hardware elements within server rooms in data centers which is costly as this would require

3.2 Presto

Presto uses manages congestion by dividing all incoming data into smaller sub-flows that then are divided evenly across the network. Each package has about equal size as a result. This leads to an more even usage of the network compared to ECMP.

It is not guaranteed that all sub-flows will arrive in order. As a result the system must account for it. This creates a small additional overhead and parts of a flow might get lost and must be resend. The method accounts for these eventualities and tries to minimise the impact but these obstacles are still more prevalent than in other methods.

Presto's load balancing is implemented in the soft edge as part of the vSwitch. This enables it to change packages without interfering with VMs. This same decision also makes it easy to implement in any existing system as it does not require any special hardware.

3.3 Clove

A major advantage to Clove is the manner of deployment. When viewed compared to CONGA, Clove is a software solution allowing it to be used on currently existing hardware rather than needing to replace hardware for its benefits. Like CONGA, not all networking needs to use the same methods as destinations that do not use Clove treat packets received from Clove-enabled devices as normal TCP packets. Another advantage to Clove is the incremental routing algorithms proposed. While the baseline Clove Edge-Flowlet algorithm routes flowlets randomly, the worst case scenario for this is to be equal to ECMP and thus already preferable. However, when the network hardware is more modern, it allows Clove to make use of more advanced features, and thus when incrementally upgrading networking hardware, Clove will become better able to deal with larger flows and higher network usage.

Unfortunately, in the best case scenario, Clove will achieve about 95% the performance that CONGA is able to achieve, combined with the fact that this best case scenario is only achievable on the newest networking equipment available, it currently seems like using Clove is similar to CONGA, where new hardware will have to be acquired to gain better performance. However, since the new hardware acquired for Clove is general hardware that can be used with other methods down the line, and CONGA specific hardware is limited to CONGA systems only, Clove is still preferred.

3.4 W-ECMP

W-ECMP uses the maximum congestion along each potential path between two leaf nodes to divide the incoming traffic evenly according to current load of the network. The information is collected by adding a header to other packages, keeping network overhead to a minimum.

The implementation of W-ECMP happens inside the data-plane. The paper realises this by using the programming language P4 since it is not bound to specific protocols or hardware. It still does require switches that are compatible with P4.

W-ECMP splits incoming data flows into flowlets. These flowlets will arrive in the right order and do not need to be reordered.

The paper compares W-ECMP to ECMP and Hula. A method the authors claim to be an improved version of Presto. They data demonstrates that W-ECMP is almost as fast as ECMP for small flow sizes (¡100kB). Where Hula is much worse. In return it is comparable or even slightly better than Hula when handling large flows(¿10MB).

The paper has a weakness in that it does not make clear how its solution would be implemented in a more complex example. In the paper the network has two layers of two switches each. For each layer one switch is assigned a one or a zero. The path trough the network is then described by a series of bits. The paper does not describe how it would handle a network with more than two possible paths per step.

3.5 CONGA vs Clove

While in older network balancing research the ECMP method is mostly used as a base case to compare to, the paper on Clove makes use of CONGA as a base to compare to. This shows that, while expensive in implementation, CONGA seems to gain much speedup over ECMP, as it is considered a better test rival in the Clove paper than ECMP. There are two major differences between CONGA and Clove. The first of these differences is in what the load balancing mechanism reacts to, and the second difference is the manner of deployment.

The first difference, being the manner of deciding how to balance a load, is a minor difference between CONGA and Clove. While CONGA makes use of congestion metrics that are piggybacked onto the data packets being sent, Clove makes use of mechanisms built into currently existing hardware, ranging from using no metric at all, and deciding paths at random, to using In-Network Telemetry that allows Clove to react proactively to congestion. Unfortunately, even with the the best and newest networking equipment Clove is not able to match the performance of CONGA. However, when a network that makes use of CONGA does not contain all networking equipment that supports CONGA, the method falls back to using ECMP, for which it is

shown that Clove can perform much better, like achieving more than four times the performance at 80% network load.

The second difference between CONGA and Clove comes into play when networking hardware is taken into account. For CONGA to gain its efficiency that it is able to reach, specialized proprietary networking equipment is needed. Compared to Clove, that is able to make use of existing hardware, by the fact of it being a software based solution, Clove seems to have a much bigger chance at being implemented at large scales. Even though CONGA is able to outperform Clove, even in Cloves best case scenario, it seems that CONGA is better used in niche use cases, such as in the research and development field, as it is still a very efficient method, outperforming Clove. In a more general case, Clove is a much better fit for large scale usage, as it is able to improve currently existing hardware in three different levels, with the best level being on hardware that is currently already five years old.

3.6 Presto vs W-ECMP

Both Presto and W-ECMP approach the issue of congestion from different perspectives. Presto operates on the edges of the network, while W-ECMP measures and operates inside the network. These methods are difficult to compare directly as a result.

They share the fact that they are supposed to be implementable on existing infrastructure. Presto uses vSwitches and Hypervisors for its functionality. Elements that have been implemented in modern operating systems for some time already. W-ECMP instead is implemented in the data plane of the network using the p4 language. It can operate independently of network protocols and hardware, as long as all the switches in the network support the p4 language. As a result we will give Presto an edge in universal applicability.

A significant difference between them is how both methods handle the state of the network. Presto is not aware of the current state of the network. It only gets updates when significant events, such as a node failing happen. W-ECMP is instead regularly receiving updated about the level of congestion along all paths through the network.

3.7 Presto vs Clove

The main difference between Presto and Clove is in which OSI layer the method is active. While Presto is active in the higher level application layers, Clove is can be found working in the transport layer. This difference can explain why Clove is much better in asymmetric networks, or in networks with link failures, as Presto is unable to make use of any congestion or asymmetry metrics into account. The main reason that Presto is used is for its simplicity. A server is able to deploy a Presto cluster that allows all nodes within that cluster to communicate using the Presto method. However, this is not possible in some cases, like when using high-performance network attached storage servers that bypass the application layer all together. This means that in a simpler system, that does not requires high speed storage servers Presto is a good candidate for use, as it is much simpler to make use off. However, in the cases where more server specific application are in use that bypass OSI networking layers, like with high-performance network storage, Clove is not only the better choice, Presto does not get to be picked in this case as it is not possible to make use of this.

4 Conclusion

In our paper we compared four different approaches for how to balance the traffic inside a data center network. Each paper proposes a unique way of improving on the current standard, ECMP. Conga stood out as it was referenced in the other three papers, alongside an issue with it that is solved in the presented method.

Overall Clove seems to be the most promising generalist that could be implemented in the most situations. Every other method has restrictions that exclude it from some applications. The second most likely seems to be W-ECMP as it is only limited by the usage of already versatile hardware.

5 Future Work

A possible future project would be to run these methods on the same hardware to collect easier to compare data. This was not feasible in the frame of this paper as it would require more significant resources than we have available.

References

[1] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, SIGCOMM '08, page 63–74, New York, NY, USA, 2008. Association for Computing Machinery.

[2] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, and G. Varghese. Conga: distributed congestion-aware load balancing for datacenters. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, page 503–514, New York, NY, USA, 2014. Association for Computing Machinery.

[3] Cisco. Cisco nexus 9000 series nx-os programmability guide, release 9.2(x). https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/92x/programmability/guide/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x_chapter_0100001.html, 2019. Accessed: 2024-02-29.

[4] Cisco. Cisco catalyst 1000 series switches data sheet. https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-1000-series-switches/nb-06-cat1k-ser-switch-ds-cte-en.html, 2023. Accessed: 2024-02-29.

[5] P. Garg and Y.-S. Wang. NVGRE: Network Virtualization Using Generic Routing Encapsulation. RFC 7637, Sept. 2015.

[6] P. Gill, N. Jain, and N. Nagappan. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, page 350–361, New York, NY, USA, 2011. Association for Computing Machinery.

[7] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. Vl2: a scalable and flexible data center network. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, SIGCOMM '09, page 51–62, New York, NY, USA, 2009. Association for Computing Machinery.

[8] S. Kandula, D. Katabi, S. Sinha, and A. Berger. Dynamic load balancing without packet reordering. *SIGCOMM Comput. Commun. Rev.*, 37(2):51–62, mar 2007.

[9] N. Katta, A. Ghag, M. Hira, I. Keslassy, A. Bergman, C. Kim, and J. Rexford. Clove: Congestion-aware load balancing at the virtual edge. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '17, page 323–335, New York, NY, USA, 2017. Association for Computing Machinery.

[10] M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright. Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks. RFC 7348, Aug. 2014.

[11] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley. Improving datacenter performance and robustness with multipath tcp. *SIGCOMM Comput. Commun. Rev.*, 41(4):266–277, aug 2011.

Analysis of Three Turing Award Laureates

Hayo Riem and Jessica Buscop

Abstract— The Turing Award is an annual prize for contributions of major technical importance to the field of computer science. It is widely recognized as the highest distinction in the field, often referred to as "The Nobel prize of computing". This paper will perform an qualitative analysis on the seminal lectures and contributions of three Turing award laureates: Edsger Dijkstra, Robin Milner, and Leslie Lamport, with the aim to answer the question: "What are the commonalities between the lectures and contributions of Edsger Dijkstra, Robin Milner, and Leslie Lamport?". The findings of this analysis show several commonalities in the work of the laureates, such as the discussion of foundational concepts. Moreover, the laureates show that computer science is a field intertwined with multiple other fields, such as mathematics and engineering. These assessments of the seminal lectures provide insightful perspectives on the contributions that have significantly impacted the field of computer science.

Index Terms—Turing Award, computer science, Dijkstra, Milner, Lamport, analysis, commonalities, contributions, concurrency.

1 Introduction

The Turing Award is the highest honor in the field of computer science, akin to the Nobel Prize. Named after the influential mathematician and computer scientist Alan Turing, it recognizes individuals who have made groundbreaking contributions to the field. Administered by the Association for Computing Machinery (ACM), this prestigious award celebrates scientists whose work has significantly advanced the theory and practice of computing.

Our paper aims to analyze three Turing Award laureates: Edsger Dijkstra, Robin Milner, and Leslie Lamport. Each of the laureates gave a lecture after winning the Turing Award, in which they described the field of computer science, while also talking about their contributions. By delving into their lectures, as well as their contributions, our aim is to provide a comprehensive understanding of the innovative ideas and methodologies presented by these scientists, shedding light on their impact on the field.

With this analysis, we want to answer the following research question: "What are the commonalities between the lectures and contributions of Edsger Dijkstra, Robin Milner, and Leslie Lamport?" By investigating their contributions, we will gain a better understanding on the evolution of Computer Science. Extracting common themes across these contributions, but also the scientist's perspectives in their lectures, can provide valuable insights about the way they all shaped the field and what they deemed important.

2 Approach and Methodology

To find commonalities between the described work of the three Turing Award laureates, we will examine the individual seminars of Dijkstra, Milner, and Lamport, describing the state of the field at the time and presenting their contributions to it. The lectures will be used as a starting point, extracting information about the state of the field, about their contributions, but also about their thought processes and ideas. This will be complemented by information of other sources to provide a complete picture of the state of the field, and the contributions that advanced this state. Next, we want to identify common themes across the lectures and contributions of these Turing laureates. We will use the previously obtained findings, together with the full content of their lectures, to understand what the lectures and the contributions of the scientists have in common. We will provide our interpretation on these themes and describe how we think this knowledge can be used.

Hayo Riem is a MSc Computing Science student at the University of Groningen, E-mail: h.p.riem@student.rug.nl.
Jessica Buscop is a MSc Computing Science student at the University of Groningen, E-mail: j.r.buscop@student.rug.nl.

3 Lecture of Dijkstra

Dijkstra, born in the Netherlands in 1930, was a mathematician and computer scientist renowned for his work in algorithm design, programming languages, and formal methods. Educated at the University of Leiden, notable achievements leading up to his Turing Award include the development of Dijkstra's algorithm and contributions to structured programming.

3.1 Context

Edsger Dijkstra provides a reflection on the state of computer programming in his Turing Award lecture [5]. He also speaks about his vision of the future of software development and advocates for a fundamental shift in approach, to address the growing complexity of software systems.

3.2 State of Computer Science

In his lecture, Dijkstra describes the state of computer science from two sides: the hardware side, and the software side. In his brief historical overview, he describes the relation between the first electronic computers and the programmers. As it was obvious that the machines would have a limited lifetime, programmers knew that little of their work would have lasting value. Moreover, programmers were limited by the specifications of the machine. At the time, programming was seen as nothing more than optimizing the efficiency of the computational process, as the machines had little memory and were slow. This led to the naive expectation that once more powerful machines were available, programming would no longer be a problem. However, in the next decades, it became apparent that something different happened: a software crisis where programming problems became larger as the machines became more powerful. The major cause of this crisis, according to Dijkstra, is that the machines have become several orders of magnitude more powerful. The better, and more reliable, hardware allowed the feasibility of solutions that were previously impossible, and these solutions now needed to be programmed.

The Software Scene

As better hardware became available, the complexity of software systems started to grow at a rapid speed, outpacing the advancements in hardware. To describe the state of the software scene, Dijkstra mentions five different software projects: EDSAC, FORTRAN, LISP, ALGOL60, and PL/1. The various projects illustrate the diversity of approaches and outcomes in software development. Each of these projects represents a different approach to programming and software design, some providing good value while others showed significant shortcomings. Because of the rapid development of software systems, a lot could be learned from both the positive contributions as well as the mistakes that were made. This state of programming enabled new

visions as for how the future of programming could look like, and how better programs could be made.

3.3 Contributions

One such vision was described by Dijkstra: "We shall be able to design and implement the kind of systems that are now straining our programming ability, and that besides that, these systems will be virtually free of bugs." The programming process will become cheaper if programmers discover the means to avoid bugs. Similarly, programmers will be more effective if they do not have to waste their time debugging, so they should not introduce bugs to begin with. While this vision may seem too good to be true, Dijkstra describes three major conditions that must be fulfilled for this revolution to happen: The world must recognize the need for the change; secondly the economic need for it must be sufficiently strong; and, thirdly, the change must be technically feasible.

1. *Recognizing the need* With the open admission of the software crisis, it became generally recognized that the design of a large and sophisticated system is going to be a difficult job.
2. *Economic need* In the current situation (at the time of the lecture), the price to be paid for the development of the software is of the same order as the price of the needed hardware, which is more or less accepted. However, hardware prices are likely to decrease which would disrupt the current balance, and society will not accept this. Therefore, it is important that programmers learn to program more effectively
3. *Feasibility* According to Dijkstra, the vision is technically feasible as long as the right approaches, tools, and thought processes are used. He highlights the importance of constructive approaches to program correctness, and using effective abstraction to compose programs. Furthermore, he emphasizes the importance of simplicity and systematicity in programming languages, as well as the use of hierarchy and factored solutions in solving complex problems.

Dijkstra has made several contributions that support the vision as described in his lecture, which laid the groundwork for methodologies and techniques aimed at improving software reliability and correctness.

Structured Programming

Structured programming is a programming paradigm aimed at improving the clarity, quality, and development time of a computer program. Edsger Dijkstra was a vocal advocate of this concept and has provided several writings and lectures emphasizing his views on this topic. An often cited example is a published paper "Notes on Structured Programming" [2], where Dijkstra advocates for the use of structured control flow constructs like loops, conditionals, and subroutines, which make program logic more explicit and manageable. He also emphasizes the importance of program correctness and argues that structured programming contributes to the creation of correct programs. The paper encapsulates his vision for improving programming methodology by using structured programming principles. It played a crucial role in shifting the mindset of the programming community towards more disciplined practices and influenced the design of programming languages.

Formal Methods

Another concept that Dijkstra deemed important, was that of using formal methods to enable rigorous analysis of software correctness, allowing developers to identify and eliminate bugs at the design stage itself. Unlike informal methods, which may rely on human intuition and informal reasoning, formal methods offer a systematic approach to software design and verification based on mathematical principles. Dijkstra was concerned with the correctness of software systems, and thus advocated for the use of formal methods as he believed these provided a level of precision and rigor necessary for ensuring correctness and reliability of software systems. He published several writings highlighting the importance of formal methods, as well as providing formal approaches to problems [6, 4].

Dijkstra's Algorithms

While his vision, and his advocacy for various concepts that aligned with this vision, have had a major impact on the field of computer science, Dijkstra is also known for the development of several algorithms. A well-known example is 'Dijkstra's algorithm', a graph search algorithm used to find the shortest path between nodes in a graph, but he also developed the 'Banker's algorithm', a resource allocation and deadlock avoidance algorithm used in operating systems. Another concept introduced by Dijkstra is the semaphore mechanism, a synchronization primitive for concurrent programming. All these developments reflect his vision by prioritizing clarity and simplicity, providing formal mechanisms, as well as taking system reliability and robustness into account.

3.4 Lessons for the future

A clear lesson that Dijkstra wanted to teach was to embrace humility. He emphasized the importance of recognizing limitations of one's own ability and advocated for a humble approach to software development, acknowledging that writing correct, reliable software is a challenging task that requires discipline, rigor, and continual learning. Another important take-away of Dijkstra's lecture is his emphasis on clarity and simplicity in program design and implementation. Written code should be easy to understand, maintain, and debug. Lastly, Dijkstra encouraged programmers to continuously improve their practices and methodologies, by striving for disciplined approaches and valuing correctness and reliability.

4 Lecture of Milner

Milner, born in England in 1934, was a computer scientist acclaimed for his work in concurrency theory, formal methods, and programming languages. Educated at King's College, Cambridge, notable achievements leading up to his Turing Award include the development of the π-calculus and his work on the Calculus of Communicating Systems (CCS).

4.1 Context

In his Turing Award lecture, Milner particularly talks about a thread of interest that has preoccupied him for over two decades: the semantic basis of concurrent computation. Milner emphasizes the importance of understanding concurrency mathematically. Concurrent models help us because they make sense of complex scenarios involving simultaneous interactions, like information flow in an insurance company, network communications, real-time communication among in-flight control computers, managing concurrency in databases, understanding how parallel object-oriented programs behave, and analyzing variables in concurrent logic programming.

The variety of applications shows that there is no one-size-fits-all approach. Each situation may require different terms and methods for discussion and analysis. However, there is a fundamental need for a common framework to organize explanations across various levels. In traditional sequential computing, we have a common semantic framework based on mathematical functions, using λ-calculus. But for concurrent programming and interactive systems, we lacked a comparable framework.

4.2 State of computer science

λ-Calculus

λ-calculus is a foundational formalism in computer science for expressing computation using functions. This simple yet powerful formalism serves as a basis for understanding computation and forms the foundation of functional programming languages. Milner was heavily inspired by λ-calculus for his later work. In λ-calculus, the syntax comprises three fundamental components[1]:

1. *Lambda Abstraction* defines a function by introducing a parameter followed by a dot. For example, in $(\lambda x.P), (\lambda x.)$ denotes the beginning of a function definition where x serves as the parameter, and P the body of the function.

2. *Variables* represent values or arguments that can be passed to functions. In λ-calculus, variables are placeholders for values.

3. *Application* applies a function to an argument. It follows the form of a function followed by an argument enclosed in parentheses. For example, in $(\lambda x.P)\,(y)\,x \rightarrow P[x := y]$ which represents that in the body of function P, x gets the value of y.

In Milner's paper, he praises λ-calculus for its effectiveness in modeling sequential formal languages. He acknowledges its remarkable versatility, with the ability to seamlessly accommodate various data structures like arrays, lists, and trees.

However, Milner argues that in a basic calculus, retaining sequential composition while adding parallel composition may lead to redundancy and violate the principle of homogeneity. Milner illustrates that in λ-calculus, there is no direct way to represent communication between concurrent processes. This lack of communication primitives hinders the accurate modeling of concurrent systems. Similarly λ-calculus struggles to model concurrency itself, as it is inherently tailored for sequential computation. Moreover λ-calculus lacks support for process mobility, a crucial aspect of many concurrent systems.

Milner's insights underscore the importance of developing formalisms tailored specifically for concurrency, rather than merely extending existing sequential models.

Calculus of Communicating Systems

One of Milner's contributions was the development of the Calculus of Communication Systems (CCS), a formal language for describing concurrent systems. However, in his discussion, Milner found CCS lacking because it primarily focused on extending sequential constructs to handle concurrency, rather than providing a fundamentally new formalism tailored specifically for concurrent systems. Milner argued that this approach did not adequately capture the essence of concurrency. It risked diluting the simplicity and homogeneity of the model by introducing redundancies and complexities.

Milner provides an example for CCS, depicted in Figure 1. Using this example, Milner critiques the complexity of the CCS definition for a process called "Double," which receives a number on channel "a" and outputs twice that number on channel "b". He notes that the definition contains a variety of elements: processes (e.g., Double), channels (a, b), variables (x), and value expressions (2 × x). Milner observes that four out of these five elements (excluding operators) are already present in the basic computation rule of CCS. Milner questions whether this level of complexity is excessive, especially for a foundational calculus. He argues that while such complexity might be acceptable in a high-level programming or specification language, where users expect a rich toolkit, it is less desirable for a basic calculus.

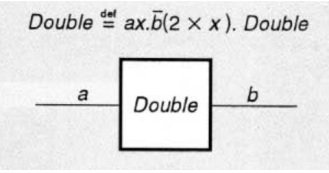


Fig. 1. Milner's doubling example [15]

4.3 Contributions

One of Milner's seminal contributions within the framework of CCS is the introduction of π-calculus, an updated version of CCS aimed at modeling concurrent computation involving dynamic communication networks.

π-Calculus

The key innovation in π-calculus lies in the notion of naming or reference. In practical computing, naming is pervasive and takes various forms, such as variables, addresses, pointers, channels, etc. These different forms serve as means of access to different entities within a system.

π-calculus introduces a shift from conventional models like CCS by emphasizing the atomic nature of interactions facilitated by naming. Each interaction step in the π-calculus involves the transmission of names, which serve as access points rather than complex data structures. This simplicity ensures that computation steps are truly atomic, enhancing the clarity and tractability of the calculus.

It also introduces dynamic communication by allowing processes to create and use communication channels on the fly, enabling more flexible and adaptive interactions between concurrent processes.
The syntax of a π-calculus process is represented as follows:

$$P ::= \quad 0 \mid \bar{x}y.P \mid x(y).P \mid P|Q \mid P_1 + P_2 \mid !P \mid (\nu x)P$$

where respectively:

- 0 represents the terminated process
- $\bar{x}y.P$ represents sending a message y on channel x, followed by the continuation of process P.
- $x(y).P$ represents receiving a message on channel x, binding it to y, and continuing as process P.
- $P|Q$ represents the parallel composition of processes P and Q, allowing for concurrent execution.
- $P_1 + P_2$ represents the non-deterministic choice between processes P_1 and P_2
- $!P$ denotes replication, where process P is replicated indefinitely, enabling repeated interactions.
- $(\nu x)P$ creates a new channel x, which is locally scoped within process P.

Example 1: Communication over multiple channels [1] We start with multiple parallel processes, channel x is only known by the first two.

$$\nu x(\bar{x}z.0 \mid x(y).\bar{y}x.x(y).0) \mid z(v).\bar{v}v.0$$

The first process sends z over channel x. The second process receives this and binds z to y.

$$\nu x(0 \mid \bar{z}x.x(y).0) \mid z(v).\bar{v}v.0$$

Now the second and third process can communicate. Because the local name x has been send to the third process, the scope is extended and it becomes the following:

$$\nu x(0 \mid x(y).0 \mid \bar{x}x.0)$$

After this it reduces to:

$$\nu x(0 \mid 0 \mid 0)$$

Example 2: Car radio Milner gives a good example of π-calculus in his paper, about a car talking to radio stations and being able to switch channels when moving [15]. This is depicted in Figure 2.

[1] Example taken from: https://en.wikipedia.org/wiki/pi-calculus

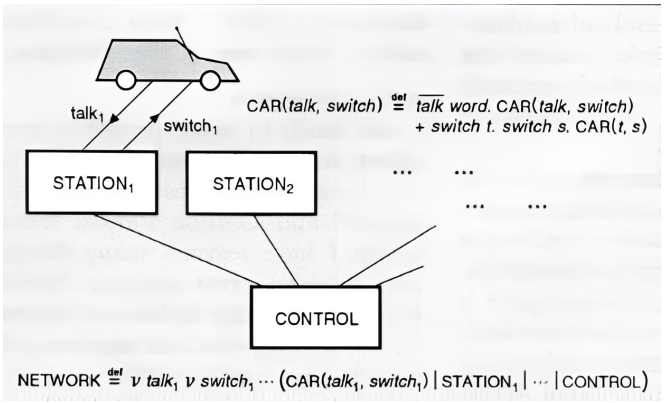


Fig. 2. Milner's car and radio example [15]

The figure illustrates the definition of the recursive CAR process, which depends on two channels: talk and switch. The CAR can perform two alternative actions: talking or switching its channels upon request from the STATION. After it talked or switched, it is called again and the process repeats. Additionally, the figure presents an expression defining the entire network. Which creates the channels talk and switch, and makes all the processes run in parallel.

4.4 Lessons for the future

Milner's paper provides valuable insights into the nature of concurrency and the challenges it presents. By emphasizing the need for formal models tailored specifically for concurrent systems, we can learn to approach the design and analysis of such systems with greater precision and clarity. Understanding the limitations of existing models, such as lambda calculus and CCS, encourages the development of new formalisms.

In the future, we can apply Milner's principles to advance various fields, including computer science, distributed systems, and network protocols. By prioritizing simplicity, completeness, and composability in our models, we can build more robust and scalable concurrent systems. Additionally, embracing naming and dynamic communication primitives, as introduced in the pi calculus, can enable us to design more flexible and adaptable systems capable of handling complex interactions.

5 Lecture of Lamport

Lamport, born in New York City in 1941, is a computer scientist recognized for his contributions to distributed systems, formal verification, and concurrent algorithms. Educated at MIT and Brandeis University, notable achievements leading up to his Turing Award include the development of logical clocks and pioneering work in formal methods for system verification.

5.1 Context

In Lamport's Turing Award lecture, the principles and evolution of concurrency in computer science are discussed [14]. His focus is on the early years, providing a personal perspective on the development of key concepts and algorithms regarding concurrency.

5.2 State of computer science

According to Lamport, the computer science of concurrency began when Edsger Dijkstra introduced the mutual exclusion problem. This was described as the problem of synchronizing processes, satisfying the requirements that no two critical sections are executed concurrently (mutual exclusion) and if a process is waiting to execute its critical section, eventually some process will execute its critical section (livelock freedom). With the introduction of the problem, Dijkstra provides a possible solution, together with a careful proof for his solution [3]. Although his implementation is correct, it does not guarantee fairness: there is no mechanism to ensure that processes enter the critical section in a fair order. Additionally, his solution may not scale well and can lead to performance problems in large-scale distributed systems.

Synchronization

Over time, various other solutions have been published, including one invented by Lamport, which will be discussed in a later section. However, synchronization remained an ongoing challenge in the field of concurrent and distributed systems. Synchronization encompasses more than just achieving mutual exclusion; it also involves coordinating the activities of multiple processes or threads to ensure correct and coherent behavior. Synchronization challenges include aspects such as deadlocks, where two or more processes are unable to proceed because each is waiting for the other, race conditions, where the outcome of a program depends on relative timing, and starvation, where processes are denied access to resources they need, despite being ready to proceed. Each of these challenges required further advancements that were not yet made at the time.

Proofs

The importance of correctness proofs is highlighted by Lamport, as the second mutual exclusion algorithm to be published, was actually incorrect [7]. The proof that Dijkstra gave with his solution for mutual exclusion was built on a computation model, in which an execution is represented as a sequence of states. Lamport defines this as the standard model, as he considers it the most generally useful model of computation. Many correctness proofs afterwards have been based on this standard model. However, as these proofs assume atomic transitions between states, there is no consideration for the temporal ordering of events. In many of the traditional proofs, reasoning about concurrent behavior often relied on informal arguments and lacked the formalism needed to reason rigorously about concurrency and distributed systems. This made it difficult to identify and address corner cases, but also hindered the development of systematic verification techniques for ensuring the correctness and reliability of concurrent algorithms.

5.3 Contributions

Lamport has made many contributions in the field of concurrency and distributed systems, and his breakthroughs have largely shaped the field as we know it today. He talks about his first major contribution in his lecture: the bakery algorithm [9]. He describes this algorithm as the first "real" solution for the mutual exclusion problem. It simulates a protocol used in retail shops, where customers take successively numbered tickets and the customer with the lowest-numbered ticket is served next. Translating this into programming, each process is assigned a unique "ticket" number based on its order of arrival at a shared resource. Processes compete for access to the resource by comparing their ticket numbers, with lower numbers having priority. This approach ensures fairness by granting access to the resource in the order of arrival, preventing starvation and guaranteeing that all processes eventually gain access to the critical section. The bakery algorithm marked a significant advancement in concurrency control algorithms, providing a practical and efficient solution to the mutual exclusion problem that addressed the shortcomings of earlier approaches, such as Dijkstra's solution.

Logical Clocks

Following the previously described synchronization problems, we come to another significant contribution made by Lamport. In distributed systems, it can be hard to say that one of two events occurred first. The concept of one event happening before another defines a partial ordering of events, and this was a concept that was not understood well in the context of distributed systems. Establishing a consistent global ordering of events posed a challenge, coming with problems such asynchrony, uncertainty, and relying solely on physical clocks. In 1978, Lamport introduced the first implementation of a logical clock [10]. Logical clocks provide a mechanism for ordering events in distributed systems based on their causal relationships.

While specific implementations may vary, the concept of logical clocks still plays a major role in modern distributed system.

Formal Proofs

Another part of concurrency that was highly impacted by Lamport's contributions, is regarding proofs for concurrent and distributed algorithms. Lamport advocated for the use of formal verification techniques, such as theorem proving, to verify the correctness and reliability of both concurrent and distributed systems. He published several papers providing novel proof techniques and exploring various approaches for proving the correctness of programs [11, 12]. In addition to advocating for the use of correctness proofs, Lamport also introduces the field to TLA+: a formal specification language for describing system behaviors and reasoning about concurrent and distributed systems [13]. TLA+ allowed engineers to describe complex systems in a clear and understandable manner, and its intuitive syntax and semantics ensured widespread adoption. The introduction of TLA+ stimulated research and development in the areas of formal methods, model checking, and software verification. Nowadays, it is still used both in academia and industry [16, 8].

5.4 Lessons for the future

From Lamport's lecture, as well as the contributions he has made to the field, we can extract several key points. First of all, concurrency is a fundamental concept. Lamport highlights various challenges and has made major contributions that advanced the field of concurrency and distributed systems. As computing systems become increasingly parallel and distributed, concurrency will continue to be a critical area of study and innovation. Another concept that Lamport emphasizes in his lecture, and which we also see back in his contributions, is the importance of formal methods, such as temporal logic and formal verification. Future advancements in computer science could benefit from continued research in formal methods for ensuring system correctness and reliability. Finally, Lamport's lecture serves as a reminder of the need for innovation and exploration. The history given of concurrency combined with his contributions to the field exemplifies how significant advancements are important to develop the field of computer science.

6 Analysis & Implications

Using the analyses of the previous sections, we will now outline commonalities between the lectures and the contributions of the laureates. Together with these findings, we will provide our interpretation of the relevancy of these common themes and how they could be applied in modern computer science.

Foundational concepts

Milner's lecture introduces the π-calculus, a formal framework for describing concurrent processes and their interactions. Dijkstra introduced a vision that would drastically change the field of programming, and advocated for various concepts that supported this vision. Lamport changed the field of distributed programming with his invention of logical clocks and the invention of TLA+. All these contributions laid the groundwork for developments that are now used in modern programming.

A lot of the work of the laureates define foundational concepts that are still being used nowadays. By acknowledging the significance of these contributions, the field of computer science can foster a culture of innovation that builds upon existing knowledge and pushes the boundaries of what is possible. The insights and perspective provided can help programmers understand how to make the next major breakthroughs in the field.

Formal models

The need for formal approaches is emphasized across all lectures. All three laureates made significant contributions towards the development and adoption of formal models. As mentioned before, π-calculus provides a formal model for describing and analyzing concurrent computation. Dijkstra actively advocated for the use of formal methods to improve the analysis of software correctness, and also developed several formal solutions to problems. Lamport highlighted the importance of the use of formal models as well, and introduced TLA+, a formal specification language for describing system behaviors.

These findings emphasize the need to use formal models to aid programmers in writing correct software and verifying its reliability. As Dijkstra also describes in his lecture, using formal approaches can help prevent the introduction of bugs, which can lead to a more efficient programming process.

Historical perspectives

All three lectures provide historical contexts or perspectives. Dijkstra hints at future development, doing so by reflecting on the past state of programming. He uses these perspectives to shape a vision on what can be improved to create a more efficient and abstract manner of programming. Lamport explicitly discusses the history of concurrent programming, while highlighting important approaches. Milner focuses on the π-calculus and situates it within the context of historical developments in concurrency theory.

When trying to invent new solutions, it is important to take the history into account. Understanding the patterns on how similar problems were solved in the past, can help programmers think in the right direction, but also help avoiding mistakes that have already been made. Acknowledging which approaches worked and which did not can provide a more clear image on the problem, as well as the solution space. This can also lead to more efficient programming.

Concurrency

Concurrency emerges as a recurring theme across the Turing Award lectures of Dijkstra, Milner, and Lamport. While each laureate addresses concurrency from distinct angles, they all acknowledge its significance in computing. Edsger Dijkstra implicitly talks about the challenges of concurrency by advocating for disciplined approaches to program correctness and reliability. He sets the foundation for handling complexity in concurrent systems. In contrast, Robin Milner's π-calculus illustrates the dynamic nature of communication and interaction in distributed computing environments. It highlights the necessity of formal frameworks tailored for concurrent computation. Finally, Leslie Lamport's work in distributed systems emphasizes the importance of concurrency in ensuring the correctness and reliability of complex systems. Through the development of algorithms such as the bakery algorithm and logical clocks, Lamport provides practical solutions for managing synchronization and ordering in concurrent environments.

Concurrency is an ever-evolving process, and even in modern programming it is still at the foreground of innovations. While the state of concurrency has evolved significantly, concurrency remains a complex and multifaceted problem in modern computing. The challenges described by the scientists, are often still a problem nowadays. Addressing these challenges requires a combination of theoretical insights, practical experience, and innovative solutions, so it is important that we keep our focus on concurrency while also ensuring the correctness, scalability, and reliability of concurrent and distributed systems.

Interdisciplinary connections

Lastly, each lecture shows how computer science relates to other fields. Dijkstra introduces a social aspect to programming, stressing the significance of acknowledging one's limitations and adopting a humble approach to writing code. Lamport's historical overview hints at the interdisciplinary nature of concurrent programming, which draws insights from mathematics, engineering, and other fields. Milner's introduction of the π-calculus suggests connections to mathematical logic, as well as theoretical computer science.

The interdisciplinary connections show that there is more to computer science than just programming. As Dijkstra emphasizes, the perspective of the individual programmer plays a significant role. One should know their limitations, and accept that collaboration, peer review, and continuous learning are vital components of software development too. Similarly, the connections shown in the lectures of Lamport and Milner should encourage developers to look beyond the boundaries of computer science alone, incorporating knowledge and techniques from diverse domains such as mathematics and physics.

These findings help answer the research question of this work: "What are the commonalities between the lectures and contributions of Edsger Dijkstra, Robin Milner, and Leslie Lamport?". We find that concurrency is an important theme in the field, with difficult challenges even with the advancements already made. Advocacy for formal methods is another recurring theme between the laureates, but we also find that computer science is intertwined with various other fields. This knowledge can be used as a guideline for modern computer science, to sustain efficient programming and keep improving the field.

7 Discussion

As stated in the beginning of the paper, the aim of this research was to analyze the contributions made by three Turing Award laureates, and to compare these findings to extract common themes. By incorporating various parts of the lectures given by the laureates, we described the state of the field at the time, and discussed several contributions that significantly impacted the state. These analyses show what influenced the contributions of the scientists, provide insights on the perspectives of the laureates, and highlight concepts the laureates deem important. Next, we extracted commonalities between the work of the laureates, their insights, and their lectures. The findings emphasize the relevancy of the contributions, but also expand our knowledge on the history of the field of computer science and provide us with directions to further advance the field.

While we think our approach provide a solid framework for analyzing the work of the laureates, there are a few shortcomings to be addressed. First of all, we discuss scope limitations. All three scientists made major contributions to the field of computer science, and in our research only a few are addressed, mostly just related to the content of their lecture. This may lead to oversimplifying their work, or neglecting other important aspects. Similarly, we set no specific criteria for our analysis, instead we mostly used the content of the lectures as a guideline on where to start. In future research, a clear scope with clear metrics could be defined. Furthermore, in this research we have only identified commonalities, but have not taken contrasting viewpoints into account, something that may be interesting for future work. Lastly, another aspect that was not considered was broader contextualization. For this research, we only focused on the field of computer science, while these contributions may also have influenced subsequent research and advancement beyond the confines of computer science.

7.1 Future Work

Next to the possible improvements described in the previous sections, there are other ways to enhance the knowledge on the history of computer science even more. Future research should analyze the contributions of more Turing Award laureates, possible focusing on other aspects of computer science. In this paper, concurrency was a recurring theme in the lectures and contributions, and the laureates made major contributions towards this concept. Future research could examine whether other parts of the field were also changed mainly by Turing Award laureates and whether their focus is on similar concepts such as formal methods. Additionally, the work of more recent laureates could be examined and compared to that of older laureates: do the same common themes recur?

8 Conclusion

The Turing Award is a prestigious annual prize awarded to scientists that made fundamental contributions to computer science. This work analyzed the lectures and contributions of three Turing Award laureates: Edsger Dijkstra, Robin Milner, and Leslie Lamport. Dijkstra's work outlined his vision for the future of programming, emphasizing the need for mathematical rigor, simplicity, and systematic methods to ensure the correctness and reliability of software systems. Lamport described the history of concurrent programming and we saw how his contributions shaped the field of concurrency. Lastly, Milner's lecture introduces λ-calculus, the Calculus of Communicating Systems (CCS) and the π-calculus, foundational formalisms for studying concurrency and interaction in distributed systems. Using these analyses, we identified commonalities between the work and insights of the three computer scientists. All three lectures discussed foundational concepts of programming, while using historical perspectives to emphasize the significance of their work. The use of formal proofs was highly emphasized to ensure the correctness of software. Lastly, the laureates show that advancements made do not only impact the field of computer science, but rather that there are various connections between computer science and other fields. The analysis performed in this work presents valuable insights on the evolution of computer science, by providing a comprehensive understanding on the contributions that have significantly impacted the field of computer science. The common themes extracted highlight how these findings can be used to further improve the field of modern computer science.

References

[1] A. CHURCH. *The Calculi of Lambda Conversion. (AM-6)*. Princeton University Press, 1941.

[2] E. Dijkstra. *Notes on structured programming*. EUT report. WSK, Dept. of Mathematics and Computing Science. Technische Hogeschool Eindhoven, 2nd ed. edition, 1970.

[3] E. W. Dijkstra. Solution of a problem in concurrent programming control. *Communications of the ACM*, 8(9):569, Sept. 1965.

[4] E. W. Dijkstra. A constructive approach to the problem of program correctness. *BIT*, 8(3):174–186, Sept. 1968.

[5] E. W. Dijkstra. The humble programmer. *Communications of the ACM*, 15(10):859–866, Oct. 1972.

[6] E. W. Dijkstra. Formal techniques and sizeable programs. In *Selected Writings on Computing: A personal Perspective*, pages 205–214. Springer New York, New York, NY, 1982.

[7] D. E. Knuth. Additional comments on a problem in concurrent programming control. *Communications of the ACM*, 9(5):321–322, May 1966.

[8] I. Konnov, J. Kukovec, and T.-H. Tran. Tla+ model checking made symbolic. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–30, Oct. 2019.

[9] L. Lamport. A new solution of dijkstra's concurrent programming problem. *Communications of the ACM*, 17(8):453–455, Aug. 1974.

[10] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, July 1978.

[11] L. Lamport. A new approach to proving the correctness of multiprocess programs. *ACM Transactions on Programming Languages and Systems*, 1(1):84–97, Jan. 1979.

[12] L. Lamport. An assertional correctness proof of a distributed algorithm. *Science of Computer Programming*, 2(3):175–206, Dec. 1982.

[13] L. Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.

[14] L. Lamport. Turing lecturethe computer science of concurrency: the early years. *Communications of the ACM*, 58(6):71–76, May 2015.

[15] R. Milner. Elements of interaction: Turing award lecture. *Communications of the ACM*, 36(1):78–89, Jan. 1993.

[16] S. Resch and M. Paulitsch. Using tla+ in the development of a safety-critical fault-tolerant middleware. In *2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, Oct. 2017.

Ensuring correctness of message-passing programs: Scribble and multiparty session types

Sarah Baksteen and Bob van der Vuurst

Abstract—In the field of program correctness, many methods have been developed to ensure correctness of various types of systems through theoretical foundations in mathematical logic. In particular, distributed systems have been a target for research, as communication between processes is an important backbone of many modern systems. Mathematical descriptions of this communication have been developed that capture important properties, such as preventing deadlocks and ensuring a lack of communication errors. Scribble is a description language for protocols that corresponds to the theory of *multiparty session types* (MPST), a description framework for message-passing protocols between any number of parties. The desired interactions for a protocol on a global level can be encoded using this framework, and then projected onto the point of view of a single participant in order to validate that an application conforms to the protocol. Scribble provides the necessary tools to apply this framework in practice and generate or validate programs according to globally-defined protocols.
In this paper, we provide a global overview of the theory of MPST and how it is related to Scribble. We discuss and explore the strengths of Scribble and how it can be used to develop robust protocols. Through hand-crafted examples, we investigate the capabilities of the framework, its strengths and weaknesses, and its useful properties regarding correctness and safety.

Index Terms—Multi-party session types, protocol correctness, process calculi, type safety, distributed systems.

1 Introduction

Given the popularity of cloud services and distributed computing, cloud infrastructure often relies heavily on communication with many devices to function properly. This communication should follow robust message-passing protocols to ensure that devices do not misinterpret messages and to avoid bad situations such as deadlocks.

A framework that can be used to describe these protocols is Multiparty session types (MPST), that specifies local types for each communicating participant in a given session [6]. A local type describes what a participant can send and receive based on its current state. Local types can be used to validate and monitor proper communication standards for each individual participant, as well as conformance of the session as a whole to avoid situations such as deadlocks.

Scribble is a high-level language that is built on the theory of MPST and provides an abstraction of describing application-level communication protocols without implementation-specific details [7]. Scribble also includes an API available in various mainstream programming languages for easy adoption in communication protocols. In this paper we will discuss Scribble, its language specification and the advantages of using Scribble for specifying a communication protocol. We will also give some example use cases of the Scribble language with simple protocol definitions and how it can be used to ensure reliable communication. We will also discuss Featherweight Scribble, which is a subset of Scribble that is proven to correspond to MPST [5]. Using a hand-made, concrete example of a protocol, we will illustrate the limitations of the theoretical framework of Featherweight Scribble and the workings of the static validation framework, which is closely related to the concrete implementation in Scribble.

This paper is structured as follows: We will first explain the basis of MPST in section 2 and then describe Scribble and Featherweight Scribble in section 3. We will discuss an example protocol defined in Scribble in section 4. Lastly, we will conclude our findings in section 5.

2 Multiparty Session Types

Multiparty session types model communications between processes, allowing for any number of parties to the conversation. This is an extension of the *binary session types* developed to model conversations between two processes [6]. Binary session types model conversations in a local manner: If we have processes P, Q communicating through the session a, then the type of a will model the interactions from the perspective of one of the two processes, as it is possible to construct a unique *dual* session type from the perspective of the other process.

As an example, let us look at a hypothetical interaction. In our scenario, a user sends a username and password to a server. The server then responds with either success or failure, and if the login attempt was a success, also sends the user a session token. From the perspective of the user, the conversation can be modeled as follows:

$$!\mathit{username}(\texttt{string}).!\mathit{password}(\texttt{string}).\\ (?\mathit{success}(\texttt{bool}).?\mathit{token}(\texttt{string}).\texttt{end} \,\&\, ?\mathit{failure}(\texttt{bool}).\texttt{end})$$

In this case, $!m(t)$ represents sending a message with label m and type t, while $?m(t)$ represents receiving a message with label m and type t. $\&$ represents that a choice is being offered to the server [6].

From the server's perspective, the same interaction is typed as

$$?\mathit{username}(\texttt{string}).?\mathit{password}(\texttt{string}).\\ (!\mathit{success}(\texttt{bool}).!\mathit{token}(\texttt{string}).\texttt{end} \oplus !\mathit{failure}(\texttt{bool}).\texttt{end})$$

which is the *dual type* to the type from the user's perspective. Here, $\oplus$ represents a choice being made internally. Notice that this type can be simply obtained from the other perspective by turning ! into ? and $\&$ into $\oplus$, and vice versa. It is generally possible to transform any session type into its dual type using similar correspondences. Hence, when using binary session types, it is only necessary to provide one of these two types to fully describe an interaction.

We will examine a small multiparty session type calculus described in a 2020 paper by Yoshida et al. [6]. A similar syntax to the one used above can be used to describe multiparty sessions locally. However, now the recipient or sender of a message has to be noted for each message as well. So for example, instead of $!\mathit{username}(\texttt{string})$ as written above, now the user's local type will include $\texttt{Server}!\mathit{username}(\texttt{string})$, where `Server` is the name of the server participant to the session. A full grammar of local session types for this MPST calculus [6] is given by

$$T ::= \texttt{end} \;\mid\; \&_{i\in I}\, \mathtt{p}?\ell_i(S_i).T_i \;\mid\; \oplus_{i\in I}\, \mathtt{q}!\ell_i(S_i).T_i \;\mid\; \mu\mathbf{t}.T \;\mid\; \mathbf{t}$$

Hence, we can have `end`, representing the end of the conversation, as well as an external choice of any number of messages, made by

• *Both authors are master's students at the University of Groningen, E-mails: s.d.baksteen@student.rug.nl, b.van.der.vuurst@student.rug.nl.*

a single other participant, and an internal choice of any number of messages sent to a single other participant. Furthermore, we have recursion, modeled by μ and the final entry in the grammar, which we demonstrate later.

In multiparty sessions, the entire interaction cannot be modeled by a single local type, in general. Therefore, we also use *global types*, which describe the entire interaction at once from a global perspective. The grammar for global types [6] is given by

$$G ::= \mathtt{end} \quad | \quad \mu \mathbf{t}.G \quad | \quad \mathbf{t} \quad | \quad \mathtt{p} \to \mathtt{q} : \{\ell_i(S_i).G_i\}_{i \in I}.$$

Global types model choices as a single type of expression, as the sender and recipient are both encoded. It is of course required that $\mathtt{p} \neq \mathtt{q}$; participants cannot send *themselves* messages.

As an example of all of the above, let us extend the login example from before. Suppose we have three participants, the `User`, `Server`, and the `Verifier`. The server relays the user's data to the verifier, and the verifier relays the result and token to the user. If the login attempt failed, the user can now choose to end the interaction, or retry. The global session type of this conversation would be as follows (simplifying the username and password data down to a single string for brevity):

$$\begin{aligned}
&\mu\mathbf{login}.\ \mathtt{User} \to \mathtt{Server} : \mathit{credentials}(\mathtt{string}).\\
&\quad \mathtt{Server} \to \mathtt{Verifier} : \mathit{credentials}(\mathtt{string}).\\
&\quad \mathtt{Verifier} \to \mathtt{User} : \{\\
&\qquad \mathit{success}(\mathtt{string}).\mathtt{User} \to \mathtt{Server} : \mathit{end}(\mathtt{bool}).\mathtt{end},\\
&\qquad \mathit{failure}(\mathtt{bool}).\mathtt{User} \to \mathtt{Server} : \{\\
&\qquad\quad \mathit{retry}(\mathtt{bool}).\mathbf{login},\\
&\qquad\quad \mathit{end}(\mathtt{bool}).\mathtt{end}\\
&\qquad \}\\
&\quad \}
\end{aligned}$$

Here, the workings of recursion are demonstrated: the variable **login** is *bound* by the μ expression, then later is used when the user wants to retry the entire interaction, where it refers to a recursive copy of the entire session type contained after the μ expression. In this way, we recurse until either the user successfully logs in, or decides to end the interaction after an arbitrary amount of failures.

This global type can be *projected* into local types from the perspective of any of the three participants. For example, the local type from the user's perspective is as follows:

$$\begin{aligned}
&\mu\mathbf{login}.\ \mathtt{Server}!\mathit{credentials}(\mathtt{string}).(\\
&\quad \mathtt{Verifier}?\mathit{success}(\mathtt{string}).\mathtt{Server}!\mathit{end}(\mathtt{bool}).\mathtt{end}\ \&\\
&\quad \mathtt{Verifier}?\mathit{failure}(\mathtt{bool}).(\\
&\qquad \mathtt{Server}!\mathit{retry}(\mathtt{bool}).\mathbf{login} \oplus\\
&\qquad \mathtt{Server}!\mathit{end}(\mathtt{bool}).\mathtt{end}\\
&\quad)\\
&)
\end{aligned}$$

and from the server's perspective, it is as follows:

$$\begin{aligned}
&\mu\mathbf{login}.\ \mathtt{User}?\mathit{credentials}(\mathtt{string}).\\
&\quad \mathtt{Verifier}!\mathit{credentials}(\mathtt{string}).(\\
&\qquad \mathtt{User}?\mathit{retry}(\mathtt{bool}).\mathbf{login}\ \&\\
&\qquad \mathtt{User}?\mathit{end}(\mathtt{bool}).\mathtt{end}\\
&\quad)
\end{aligned}$$

In this case, the two branches of the Verifier-to-User interactions have been *merged* into one choice in the Server's local projection of the global type, as the interaction between the Verifier and the User is not visible to the Server. This is only possible because the user sends *end* to the server before closing the interaction in both the success case and the case where the user voluntarily ends the interaction, making those cases indistinguishable to the server. If these interactions were not the same, then merging the two branches would be impossible, and the server's local session type would be undefined. In the case that a global session type cannot be projected into a local session type for every participant, the global protocol described by this session type is said to be ill-formed. A more detailed definition of both merging and projection can be found in [6], as well as a complete process calculus utilizing this type system, including syntax for defining message-passing processes and combining them into concrete multiparty sessions.

3 Scribble

Scribble is a language for describing protocols that could be considered as the practical implementation of MPSTs [2]. Scribble has the goal to provide tools for specifying communication protocols in an intuitive way with the guarantee that there are no deadlocks or ambiguous cases and with automatic detection and handling of communication errors. Scribble has an API available in many mainstream programming languages, such as Python [1], Java [3] and Rust [4].

The Scribble framework relies on a few assumptions of the networking infrastructure for a reliable communication protocol [2]:

- Message sending is a non-blocking operation;
- Messages are received in the same order as they are sent;
- Messages are never lost or altered in transmission.

In practice, the latter two assumptions can be met by using TCP with encryption for sending messages with Scribble. Most modern devices also include network controllers that can handle network traffic, which means that messages can also be sent without blocking the processor.

We will now look at an example protocol written in the Scribble language. A Scribble definition of the global and local protocols of the user login example given in section 2 is shown below. Note that only the global protocol has to be specified in the Scribble API, because the local protocols are created automatically by projecting the global protocol for each role.

```
1  global protocol Login (role User, role Server, role
       Verifier) {
2      rec LOOP {
3          Credentials(string) from User to Server;
4          Credentials(string) from Server to Verifier;
5
6          choice at Verifier {
7              Success(string) from Verifier to User;
8              End() from User to Server;
9          } or {
10             Failure() from Verifier to User;
11             choice at User {
12                 End() from User to Server;
13             } or {
14                 Retry() from User to Server;
15                 continue LOOP;
16             }
17         }
18     }
19 }
```

Listing 1. Global login protocol

```
local protocol Login at Server (role User, role Server
    , role Verifier) {
    rec LOOP {
        Credentials(string) from User;
        Credentials(string) to Verifier;
        choice at User {
            End() from User;
        } or {
            Retry() from User;
            continue LOOP;
        }
    }
}
```

Listing 2. Local login protocol (server's perspective)

In a practical application, the above would likely be a bit different: there would likely be separately defined types for both user credentials and the login session token returned by the verifier. Scribble supports type definitions in several ways, including XML schemas and types from the programming languages it has bindings to [7]. We present it here in this form to keep it as close as possible to the session type description from the previous section.

Comparing the two, we see that protocol description in Scribble has striking similarities to multiparty session type descriptions. `rec` corresponds to μ, `m(S) from A to B` corresponds to $\mathtt{A} \to \mathtt{B} : m(\mathtt{S})$, and so forth. In fact, Scribble inspired the development of multiparty session type theory [6], which attempts to formalize the ideas present in Scribble.

Scribble ensures that communication protocols are followed in two ways: With static verification and runtime checking. An overview of verification by Scribble is shown in Figure 1. In the static verification step it is checked not just whether the protocol follows the grammar of Scribble, but also whether it is well-formed, i.e. can be projected into local protocols for every participant. At runtime, the messages sent by a program can be checked against the protocol specification by the Scribble conversation monitor. This is done with a representation of the local protocol as a finite state machine (FSM) [7]. If some message is not present as an edge in the current state of the FSM, then the conversation monitor will block the message. This check is performed for both a program's inbound and outbound messages to ensure that the program follows the protocol, and to prevent any malicious attacks with intentionally ill-formed messages. Figure 1 illustrates the Scribble workflow, from defining a global protocol and validating it through projection, to implementing the protocol and validating each implementation using dynamic verification.

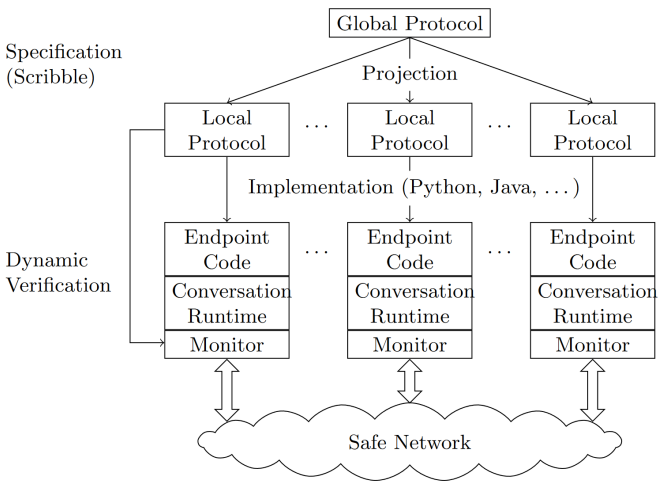


Fig. 1. The Scribble workflow, including both static and dynamic protocol verification. Image taken from [7].

In this way, Scribble provides a useful framework for checking correctness of protocol implementations, as well as validation of the well-formedness of the initial protocol, which avoids ambiguity and inconsistency [5]. The framework also serves to prevent deadlocks, because Scribble requires that all messages be sent in a particular sequence, and received before the next message is sent [7].

3.1 Featherweight Scribble

In a 2019 paper by Neykova et al. [5], a subset of Scribble, termed *Featherweight Scribble*, was presented with formal semantics, and a correspondence between it and multiparty session types was shown. Featherweight Scribble contains all the syntax elements we saw in the login protocol example: `global` and `local protocol` definitions, `m(S) from A to B` and its local equivalents, `choice at` with any number of `or` branches, `rec`, and `continue`. It also contains `end`, which signifies the end of a protocol.

Projection in Featherweight Scribble is defined recursively as follows [5]: $G\downarrow_A$, the projection of global protocol G onto participant A, is given by

$$(\mathtt{a(S)\ from\ B\ to\ C};G')\downarrow_A = \begin{cases} \mathtt{a(S)\ from\ B};(G'\downarrow_A) & \text{if } A=\mathtt{C},\\ \mathtt{a(S)\ to\ C};(G'\downarrow_A) & \text{if } A=\mathtt{B},\\ G'\downarrow_A & \text{otherwise;}\end{cases}$$

$$(\mathtt{rec\ t\ \{}G'\mathtt{\}})\downarrow_A = \begin{cases} \mathtt{rec\ t\ \{}G'\downarrow_A\mathtt{\}} & \text{if } G' \neq \mathtt{continue\ t},\\ \mathtt{end} & \text{otherwise;}\end{cases}$$

$$(\mathtt{choice\ at\ B\ \{a}_i\mathtt{(S}_i\mathtt{)\ from\ B\ to\ C};G_i\}_{i\in I})\downarrow_A = \begin{cases} \mathtt{choice\ at\ B\ \{a}_i\mathtt{(S}_i\mathtt{)\ from\ B};(G_i\downarrow_A)\}_{i\in I} & \text{if } A=\mathtt{C},\\ \mathtt{choice\ at\ B\ \{a}_i\mathtt{(S}_i\mathtt{)\ to\ C};(G_i\downarrow_A)\}_{i\in I} & \text{if } A=\mathtt{B},\\ \mathtt{choice\ at\ D}\ \bigsqcup_{i\in I} G_i\downarrow_A & \\ \quad \text{if } A\notin\{\mathtt{B},\mathtt{C}\};G_i\downarrow_A = (\mathtt{a}_i\mathtt{(S}_i\mathtt{)\ from\ D};G_i'\downarrow_A), \forall i\in I;\end{cases}$$

$$(\mathtt{continue\ t})\downarrow_A = \mathtt{continue\ t};\qquad (\mathtt{end})\downarrow_A = \mathtt{end}.$$

Here, all instances of $G', G_i, \ldots$ represent continuations of the global protocols, choices are written using the notation $\mathtt{choice\ at\ A}\,\{G_i\}_{i\in I}$ for brevity, where i ranges over some amount of branches and G_i are the contents of the branches, and $\sqcup$ is the merge operator, which, intuitively speaking, merges a number of choice branches such that the result has no duplicates and branches that start with the same message are merged into one. If merging is impossible or none of the conditions for projecting `choice at` are met, then projection fails and the global protocol is ill-formed.

We can notice that, under this definition, our login protocol example from earlier is ill-formed: it cannot be projected onto `Server` because the `choice at Verifier` branches do not continue with a straightforward message from `User`; the `or` branch continues with a `choice at User` instead. Strictly speaking, then, we would need to add another message from `User` to `Server` in this branch to make our protocol well-formed under Featherweight Scribble. The reason why this protocol was well-formed as a multiparty session type but not as a Featherweight Scribble protocol is because of the slight difference in projection operators used between the two papers: the projection operator for MPSTs is slightly more permissive because choices and single messages are treated the same for its purposes, while Featherweight Scribble projection of `choice`s requires a single message, not a choice, as a continuation.

An important reason for this difference is that in standard Scribble, choice branches do not need to start directly with a message. It is only enforced that the first message *after branching* is from the participant doing the branching. For example,

```
choice at A {
    rec LOOP {
        m1(S) from A to B;
        choice at B {...}
    }
} or {
    m2(S) from A to B;
}
```

is legal in Scribble, but clearly

$$\mathsf{A} \to \mathsf{B} : \{\mu \mathbf{t}.\, m1(\mathsf{S}).\mathsf{B} \to \mathsf{C} : \{\ldots\}, m2(\mathsf{S}).\mathsf{end}\},$$

(an attempt at writing) the equivalent MPST, makes no sense. The initial message in a choice is inherently tied to the choice itself in an MPST, so direct translation here is impossible. It is, however, always possible to unfold and flatten a well-formed Scribble protocol in such a way that a message is always the first statement in a `choice` branch, while keeping the behavior of the protocol identical. In this case, this is achieved by unfolding the first iteration of the loop to be before the loop definition. The Featherweight Scribble authors term this Scribble Normal Form (SNF), and it is an essential part of translating Featherweight Scribble protocols into MPSTs.

4 Examples

In this section, we examine the capabilities of Scribble through a concrete example of a protocol, and show in practical terms the restrictions imposed by well-formedness in the Featherweight Scribble sense.

```
global protocol Checkers(role Player1, role Player2,
    role Server) {
    choice at Player1 {
        resign() from Player1 to Server;
        loss() from Server to Player1;
        win() from Server to Player2;
    } or {
        rec TURN {
            move(Move) from Player1 to Server;
            choice at Server {
                extraTurn(GameState) from Server to
    Player1;
                continue TURN;
            } or {
                win() from Server to Player1;
                loss() from Server to Player2;
            } or {
                win() from Server to Player2;
                loss() from Server to Player1;
            } or {
                draw() from Server to Player1;
                draw() from Server to Player2;
            } or {
                nextTurn(GameState) from Server to
    Player1;
                nextTurn(GameState) from Server to
    Player2;
                do Checkers(Player2, Player1, Server);
            }
        }
    }
}
```

Listing 3. Global checkers protocol

The above is a protocol meant for a game of checkers between two players and a facilitating server. The server does all calculations, such as determining when a player gets an extra turn because they took an opponent's piece and are eligible to take another, and calculating the next game state, and therefore all actions after a player selects a move are a `choice at Server`. The `do` statement is not part of Featherweight Scribble, but in fully-featured Scribble, it is essentially a function call: in this case, we recurse over the entire protocol, swapping the roles of player 1 and player 2 each turn.

One thing to notice is that this protocol is ill-formed: in one of the branches when the server is responding to player 1's move, the server sends a message to the second player first, informing them of their win, before sending a loss message to the first player. This is in conflict with all other branches having a message from `Server` to `Player1` as their first message, and therefore any projection will fail. It is, however, not a completely unreasonable thing for the server to do: it has to send messages to both players informing them of the outcome, but the order seems immaterial in this case. This part of the protocol can be fixed simply by reversing the order of those two messages.

But we are not done: in the first branch of the same choice, player 2 is not receiving any message. This will also cause projection for `Player2` to fail, so we need to add a sort of 'hold on' message to `Player2` in that branch before the protocol becomes well-formed. We also need to add one to the beginning of the turn loop, to avoid a scenario where the `choice at Player1` statement is unable to be projected to `Player2`:

```
⋮
rec TURN {
    move(Move) from Player1 to Server;
    turnStart() from Server to Player2;
    choice at Server {
        extraTurn(GameState) from Server to Player1;
        holdOn() from Server to Player2;
        continue TURN;
    } or {
        loss() from Server to Player1;
        win() from Server to Player2;
    } or {
        ⋮
}
```

Now, let us imagine that checkers had an extra rule added to it, in which a third party acting as a referee has to make some sort of decision only in a particular situation. Intuitively, we would want to implement that case in the following way:

```
⋮
choice at Server {
    extraTurn(GameState) from Server to Player1;
    holdOn() from Server to Player2;
    continue TURN;
} or {
    requestDecision(GameState) from Server to Referee;
    decide(Decision) from Referee to Server;
    // now the server sends appropriate messages to
     the players
} or {
    ⋮
```

However, this leads to a problem: the server *has to* send a message to the same participant first in every branch. So in this case, either we have to send an extra, useless message to `Player1` in the branch with the `Referee`, or we would need to send useless messages to the `Referee` in every single other branch (in addition to the sensible messages we need to send to the referee when the game is over). In fact, the rules for projection say that every branch needs to have a message to the `Referee` in it. Otherwise, projection to `Referee` would fail. Therefore, under this system, the referee needs to be notified every turn whether or not their involvement is necessary, rather than only when they need to make a decision. Surely, this is annoying for the referee.

This shows that, in order to implement a protocol in Scribble, sometimes large changes need to be made to accommodate the restrictions of the framework before one is able to benefit from its advantages.

With a valid Scribble protocol now, however, we can give certain guarantees about any implementation. The foremost of those is that we can easily verify using Scribble's finite state machines whether or not the protocol is implemented *correctly*. This means that all steps are followed exactly as presented in the protocol. In this case, that guarantees that the game of checkers will proceed sequentially, one turn at a time, starting with Player 1's turn, then proceeding with Player 2's turn, and so forth, without any potential for confusion about whose turn it is or what message is expected next by any party. In this way, the Scribble protocol provides a precise specification for the behavior of each of the participants.

What this does not guarantee is that the rules of checkers are followed correctly. Indeed, nothing about the written protocol has any information about those rules, and Scribble cannot check correct behavior in that sense, only that global behavior conforms to the protocol. We can derive some ideas about implementation details, however. For example, it is implied by the protocol that each player's client application should check whether moves are valid before sending them, as each of the server's choices corresponds to a valid move being received.

5 Conclusion

In this paper we have discussed how MPST model communications between processes and how it expands upon binary session. We have explained the grammar through examples and demonstrated the projection of the global type to each role's local type. We have also discussed Scribble and the advantages of defining a protocol in Scribble and how the Scribble API ensures deadlock-free communication without any errors by static and runtime verification. We also looked at Featherweight Scribble and how its restrictions can make some intuitive protocols ill-formed. With a simple example of a protocol for a checkers game we have shown and explained why the protocol is ill-formed in Featherweight Scribble. We have then rewritten the protocol such that it is well-formed, but includes empty messages that do not serve a purpose and introduce overhead.

In conclusion, there are various advantages of using Scribble to ensure that no communication errors can occur by avoiding ambiguity about which message to expect at any point for any party, but it is also arguably difficult to write a complex protocol that is well-formed and also does not send empty and counterintuitive messages to comply with well-formedness constraints. The fully-fledged practical implementation of Scribble, however, is somewhat more permissive and expressive than the Featherweight version in this regard, alleviating these issues at the cost of not yet being formally proven to correspond to a useful mathematical framework.

A possible line of future work could be to validate some existing widely used protocols, such as HTTP and SMTP, and rewrite them in Scribble to check if they are well-formed. This could proof that these protocols are free of communication errors or otherwise discover a potential bug. Further, a proof that the whole Scribble language or a larger subset of Scribble corresponds to a MPST calculus would help prove the concrete benefits and guarantees of Scribble protocols.

References

[1] R. Demangeon, K. Honda, R. Hu, R. Neykova, and N. Yoshida. Practical interruptible conversations: distributed dynamic verification with multiparty session types and Python. *Formal Methods in System Design*, 46(3):197–225, June 2015.

[2] K. Honda, A. Mukhamedov, G. Brown, T.-C. Chen, and N. Yoshida. Scribbling Interactions with a Formal Foundation. In R. Natarajan and A. Ojo, editors, *Distributed Computing and Internet Technology*, volume 6536, pages 55–75. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[3] R. Hu and N. Yoshida. Hybrid Session Verification Through Endpoint API Generation. In *Fundamental Approaches to Software Engineering*, volume 9633, pages 401–418. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[4] N. Lagaillardie, R. Neykova, and N. Yoshida. Stay Safe Under Panic: Affine Rust Programming with Multiparty Session Types. In K. Ali and J. Vitek, editors, *36th European Conference on Object-Oriented Programming (ECOOP 2022)*, volume 222 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:29, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[5] R. Neykova and N. Yoshida. *Featherweight Scribble*, pages 236–259. Springer International Publishing, Cham, 2019.

[6] N. Yoshida and L. Gheri. A very gentle introduction to multiparty session types. In *Distributed Computing and Internet Technology: 16th International Conference, ICDCIT 2020, Bhubaneswar, India, January 9–12, 2020, Proceedings*, page 73–93, Berlin, Heidelberg, 2020. Springer-Verlag.

[7] N. Yoshida, R. Hu, R. Neykova, and N. Ng. The scribble protocol language. In M. Abadi and A. Lluch Lafuente, editors, *Trustworthy Global Computing*, pages 22–41, Cham, 2014. Springer International Publishing.

Implementation of Microservices: A Comprehensive Study on Design Patterns, Quality Attributes and Industry Practices

Andra Trandafir

Abstract—Microservices architecture (MSA) has gained prominence for its scalability and flexibility in software systems. Despite its popularity, a gap persists between academic research and industry practices. This paper addresses this gap through a comprehensive literature review on both academic and grey literature. Three key research questions are explored: the prevalence of design patterns, factors influencing implementation decisions and prioritisation of quality attributes. The study examines the evolution of MSA from Service-Oriented Architecture (SOA) and compares it with similar architectural styles. The findings offer valuable guidance for practitioners seeking to optimise microservices systems.

Index Terms—Quality attributes, industry, practitioners, design patterns, microservices.

1 Introduction

During recent years, microservices have gained popularity due to their scaling opportunities and the benefits they bring to businesses. Emerging from Service-Oriented Architecture design patterns [17], Microservices Architecture goes beyond the monolithic approach and component dependencies, but instead implements individual, loosely coupled modules focused on specific business tasks at a time. The final decision to implement microservices depends on the project [18, 22], because there are both benefits and disadvantages, thus sometimes the answer is to develop a traditional monolithic application instead. However, in cases where the application is in fact suited to use the MSA architectural style, which patterns would best facilitate its optimization? Although research is strongly interested in performing studies for microservices, there is a lack of available literature in this field. Furthermore, there exists a general demand in previous literature works for covering the gap between theoretical models introduced by academia and industry practices. The methodology used to address this issue is a literature review, that aims to be focused on both academia and grey literature. During the study, the following key research questions were addressed: Is there a *trend* in implementing certain patterns over the others? What are *other factors*, such as social aspects, that influence the decision of implementation? Is there a quality attribute *prioritisation* in the selection of the design pattern? The paper has the following structure: the first chapter explores the differences between different architectural styles, identifies different categories and exemplifies the communication patterns, chapter 2 presents the methodology, chapter 3 presents the analysis and the results and the last chapter is a conclusion and future recommendations.

1 Background

2.1 The Microservices Architectural Style

Microservices are a very popular design choice nowadays for architecting loosely-coupled pieces of software. The approach that is involved by this architectural style is that every business unit is deployed and run separately as an independent full-stack application. Therefore, it can be said that microservices follow a service-based application methodology [1]. Previously, only monolithic applications have existed, and continue to be prevalent, but with the several benefits that microservices bring, their widespread adoption has been notable ever since their formal introduction in 2008 by Peter Rodgers [2].

2.2 Economical Impact

The implementation of microservices in businesses like Uber, Netflix, and Amazon, which are known to be large-scale, high-traffic businesses, has been shown to offer numerous benefits, including increased agility, reduced complexity, and improved scalability [23], and can serve as an example in the decision making, especially concerning costs and business advancements. This management of microservice evolution can be challenging, and a model-based approach has been proposed to address this issue [24]. In terms of infrastructure costs, microservices have been found to be more cost-effective than monolithic architectures, with specialised services like AWS Lambda offering the potential for significant cost reductions [25; 26]. The benefits can further be seen at Uber, which has motivated their switch in 2013 in a blog post [27]. They have stated that as their product scaled more and more, from 10 to 100, then from 100 to 1000 engineers, the monoliths have started to make the team collaboration more difficult, therefore switching to microservices allowed them to advance their product in terms of scalability and quality. Furthermore, they have also specified the level of complexity they have added. The implied level of complexity in their approach is further illustrated in Image 1, and in Image 2, for Netflix and Amazon, which follow a similar decision.

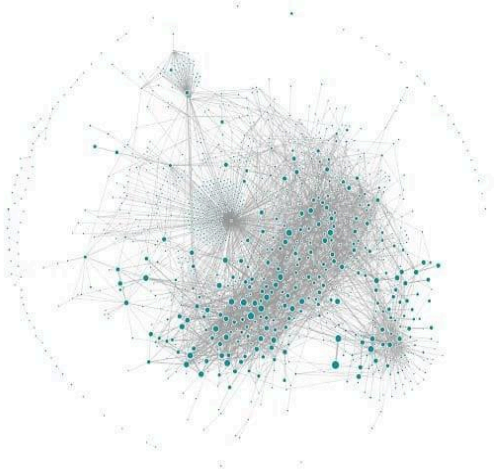


Image 1 - Uber's microservice architecture circa mid-2018 [27]

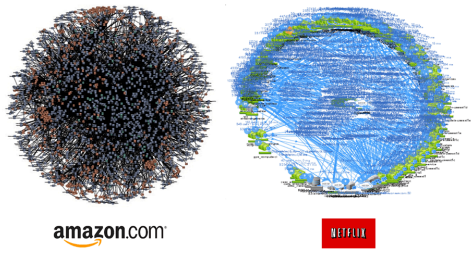
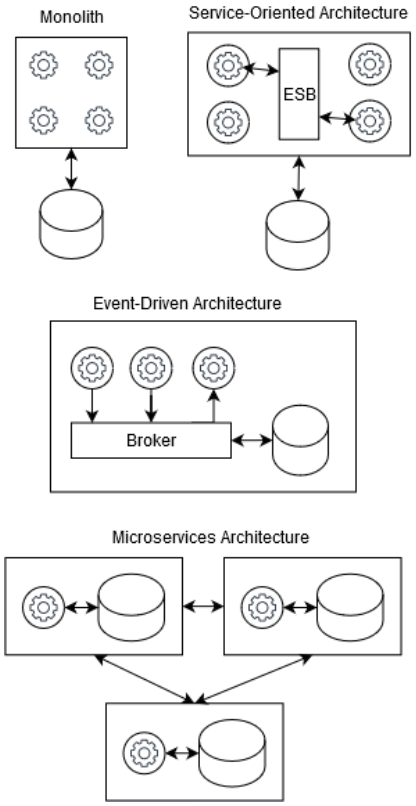


Image 2 - Microservices at Amazon (2008) and Netflix (2014) [41]

2.3 Alternative Architectural Styles

A very broad confusion can be made in making the distinction between other architectural styles, more specifically between Service-oriented architecture (SOA), Event-driven architecture (EDA) and the topic on hand, Microservices Architecture (MSA). It is therefore important to look back in the past and see where MSA has emerged from, in order to understand the key principles behind the present rationalisation. Márquez et al. [5] expanded on the comparison of monolith, SOA and MSA, while Singh et al. [27] argued about the beneficial usage of EDA in distributed systems. Figure 1 further summarises the findings of Márquez et al. [5] with the additional comparison to EDA.

Figure 1 - Different architectural styles

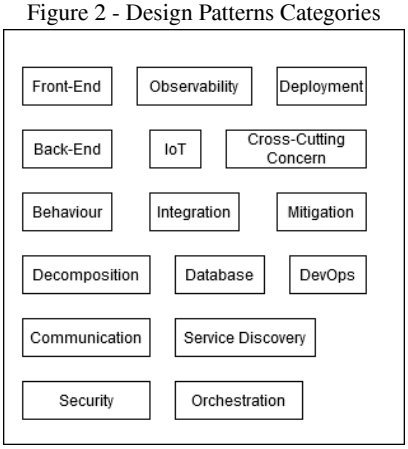


SOA is using services as well, although it is tailored for larger enterprise solutions, and has as the main characteristic an enterprise service bus, that serves as the only communication channel between the reusable modules that the style imposes. Conversely, EDA, involves the use of asynchronicity as the main characteristic: systems communicate and react to events asynchronously, enabling decoupled and scalable architectures. EDA is particularly useful for scenarios where real-time responsiveness and handling of large volumes of events are essential, such as in financial trading systems, IoT platforms, or activity monitoring and early warning systems [28, 29, 30, 31]. Finally, although MSA has evolved from SOA architectural patterns and shares similarities, their autonomy goes beyond their predecessors and provides additional levels of scalability and fault-tolerance and can be emphasised by the use of EDA [32]. Bogner et al. [3] further describe the main principles involved in SOA and MSA. MSA inherits all of the principles first presented in SOA. On the other hand, [33, 34] expand on the enhancements brought specifically by MSA: Products, not Projects - Services are seen and treated as independent entities, each with its own lifecycle management, and with a product mindset; Smart Endpoints, Dumb Pipes - the middleware between the services is simplified as much as possible, the endpoints are design with the services capabilities in mind, thus making the infrastructure as minimal and the emphasis is put on the endpoints, which are more complex and satisfy the business logic.

2.4 Patterns Categories

Waseem et al. [9] emphasised the importance of easiness for practitioners to navigate through design patterns. With their findings, along with other papers [14], [15], [5], design patterns categories have been identified, presented in Figure 2.

Figure 2 - Design Patterns Categories



2.5 Communication patterns

Communication patterns play a vital role in the design and architecture of a system [50]. They define the way in which different components of a system communicate with each other and with external parties. These patterns provide a structured approach to handle various types of communication scenarios, ensuring reliability, scalability, and maintainability of the overall system [52]. Some commonly used communication patterns include **API gateway**, which acts as a single entry point for client requests (presented in Figure 3), and the **backend for frontend** pattern, which tailors backend responses for different frontend applications (Figure 4). The **aggregator** pattern combines data from multiple sources, while the **proxy** pattern acts as an intermediary for traffic control. In addition, the **remote procedure invocation** pattern allows for the invocation of methods on remote services. Riesen [50] explains that understanding message-passing patterns in parallel applications is essential for efficient and scalable performance. The **asynchronous messaging** pattern decouples the sender and receiver of messages, enabling asynchronous communication. The **publish-subscribe messaging** pattern broadcasts messages to subscribers, while the **publish-asynchronous messaging** pattern publishes messages asynchronously. For immediate

message-response interaction, the **synchronous messaging** pattern is used, while the **asynchronous request-reply** pattern initiates requests asynchronously.

Figure 3 - The API Gateway Pattern [53]

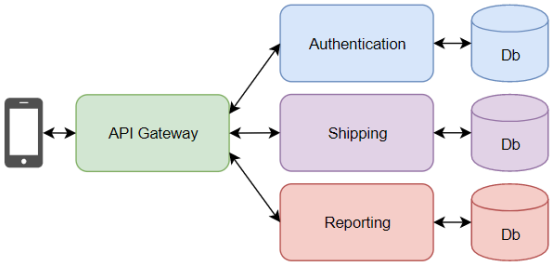
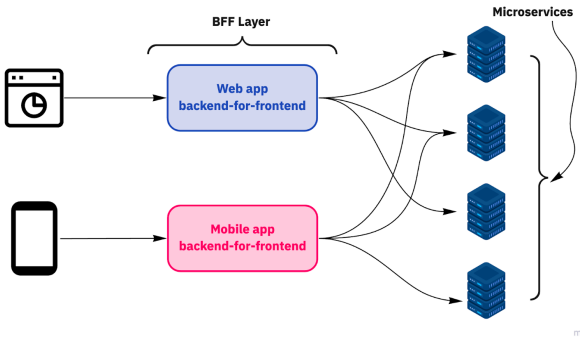


Figure 4 - Backend for Frontend Pattern [54]



2.7 Related work

Although there is a lack of literature on microservices, some studies have been conducted to address the gap between industrial practices and academic models. Dragoni et al. [19] discuss the evolution of microservices architecture and the main motivations behind its distinguishing characteristics, including its relation to object-oriented and service-oriented paradigms. In addition, Zhang et al. [20] performed a survey in industry settings to identify differences between theoretical models and practical experience, highlighting concerns with domain-driven decomposition and monitoring techniques. Quality attribute trade-offs within microservices have also been explored, with Waseem et al. [9] proposing a general approach for pattern selection that aligns with different architectural decisions. Moreover, Waseem et al. [10] conducted another study with 106 participants, revealing key metrics about the usage of microservices, including popular design patterns such as API gateway and backend for frontend for communication, and database per service for data management. Overall, these studies provide a comprehensive understanding of microservices architecture and its design patterns, serving as valuable resources for both academia and industry research.

3 Methodology

This study employed a literature review methodology, consisting of academic literature and grey literature. The study of Rocha et al. (2016) [42] was used as a guideline for including grey literature, and, as described in the study, shaped the research by encompassing informal publications, found in blogs and websites, accessed through regular search engines. Additionally, Garousi et al. [43] categorised grey literature into three levels: Level 1 involves materials with a high level of decision control and credibility, such as books, magazines, reports from government agencies, and reputable organisations in the field; Level 2 includes medium-level sources, such as annual reports, news articles, presentations, videos, and websites like StackOverflow; and finally, Level 3 comprises sources with a low level of decision control and credibility, such as blogs, tweets, and pages from social networks. Moreover, the academic sources collected were chosen based on their peer-reviewed status. Peer-reviewed papers play a critical role in the scientific community by ensuring the quality and accuracy of published work (McLellan, 2020 [45]). These papers are typically published in academic journals or presented at conferences.

3.1 Motivation

Research on microservices has gained significant attention in recent years, yet the availability of literature in this field remains limited. Previous studies have highlighted the need to create a consensus on different microservices topics, from taxonomies to implementation methodologies. The disconnect between academia and industry is a well-known and pervasive issue. Despite the growing body of research in the field of software development, there seems to be a lack of recognition for the importance of Grey Literature in this domain. Garousi et al. [43] echo this sentiment, highlighting that software development professionals produce valuable and relevant Grey Literature, which is largely overlooked by academic research.

3.2 Research questions

In order to guide the decision of implementation of microservices, and seek different opinions across different implementations, the following research questions were addressed:

RQ1: Is there a common trend in the implementation of certain patterns over others?

RQ2: What are the factors, such as different working styles, etc, that influence the decision of implementation?

RQ3: Is there a quality attribute that has a higher priority than others in the selection of the architectural pattern?

4 Analysis and Results

4.1 *Is there a common trend in the implementation of certain patterns over others?*

After examining three different research studies, which investigated academic papers and open-source projects, it has been clear that certain design patterns are more predominant.. The studies [6, 7, 8], consistently identified API Gateway as the most used design pattern. However, the rankings of other commonly used design patterns varied slightly among the studies. As pointed out by [16], the application of design patterns is heavily influenced by contextual factors, specific problems, and proposed solutions. Therefore, the frequency of using a particular design pattern depends on the recurrence of a specific context and problem in the industry. As a synthesis and alignment of the findings from the mentioned studies, API Gateway was identified as the most prominent pattern, which is further supported by other findings [38, 10, 39]. Additionally, circuit breaker, load balancer, service registry, and health check were identified as the most commonly used patterns, while proxy, discovery patterns, service instance per container, and asynchronous messaging were also frequently chosen design options. Arguably, another study has found that the most prevalent design pattern was the Proxy, followed by Saga [42]. This also concludes that the

distribution of the patterns is heavily influenced by the application constraints [16, 40].

While some studies have highlighted the general consensus that the API Gateway is the most used pattern, another study has found the Proxy pattern to be the most prevalent in their set of projects. Therefore, it can be said that the distribution of patterns is influenced by the application constraints.

4.2 *What are other factors that influence the decision of implementation?*

Osses et al. [50] discovered that professionals take into consideration the influence of various patterns on quality attributes when working on implementation, often in alignment with the pattern documentation. Moreover, there exist other noteworthy findings which can sway the decision at hand, for example the project's requirements and the technological infrastructure of the organisation [48] play a pivotal role, as certain patterns are exclusive to specific programming languages. Additionally, the expertise of the development team is taken into consideration, as improper implementation can result in maintenance complications [46]. Furthermore, constraints such as scalability requirements and security considerations [47] may also factor into the decision, while other aspects, such as feedback from previous projects, may also hold weight. Furthermore, research consistently illustrates that team dynamics and social aspects significantly impact the decision-making process for design pattern selection. [49] conducted a study wherein they analysed metrics such as team dynamics, the planning of the design process, participant actions, information gathering and sharing, approaches to analysing and comprehending the design problem, techniques for developing and adopting design concepts, as well as conflict resolution and avoidance. [13] also states that while the main goal for participants is to achieve the desired quality attributes as expeditiously as possible, there exist other considerations to ponder when selecting design patterns for microservices. The authors acknowledge that these individuals may seek more information regarding the pattern, its application, as well as potential solutions for encountered errors during implementation. Thus, it can be surmised that familiarity with a pattern may hold more significance than other factors. Ultimately, the authors conclude that the popularity of a design pattern among practitioners is not solely defined by its efficacy in achieving performance tactics, but rather by a multitude of factors that play into the decision-making process.

The popularity of a design pattern among practitioners is not solely determined by its effectiveness, but rather a combination of factors regarding the decision-making process, such as social aspects, pattern documentation, and many other factors.

4.3 *Is there a quality attribute that has a higher priority than others in the selection of the architectural design?*

The selection of architectural design for microservices is heavily influenced by various quality attributes, with scalability, flexibility, testability, performance, and elasticity being of utmost importance [35]. However, the prioritisation of these attributes lacks clear delineation. [36] and [37] both highlight scalability as a critical consideration, with the latter also emphasising the significance of performance, availability, monitorability, security, and testability. This suggests that while scalability is a key factor, other attributes such as performance and security should also be given high priority in the selection of architectural design for microservices. Conversely, a study conducted by [4] involving 106 participants revealed security as the most prominent quality attribute. This emphasis stems from the intricacies of microservices systems, which are susceptible to security vulnerabilities that can be exploited by potential intruders, as highlighted by one respondent. Following security, availability emerged as the second most crucial aspect, driven by the necessity of maintaining maximum system uptime to meet user expectations. Additionally, performance was perceived as the third most important attribute, owing to its potential to degrade due to frequent communication calls between services. Likewise, scalability was also deemed significant due to its critical role in the context of microservices architecture. Other quality attributes mentioned, in order of importance, include reliability, maintainability, usability, compatibility, testability, monitorability, functional suitability, reusability, resilience, portability, and interoperability.
However, the understanding of these attributes is still evolving, and there is a need for a comprehensive guide on quality improvement in microservices architecture.

While some studies prioritise scalability to be the first consideration in terms of quality attributes, recent research also emphasises the importance of security, availability, performance, as well as reliability, maintainability, and usability.

5 Threats to Validity

One potential issue that can undermine the validity of research studies is selection bias. This could occur because the study samples chosen for the study are not representative of the larger population, leading to limited generalizability. For example, the studies for the most prevalent pattern, although the majority showcasted the API Gateway, and others the Proxy, others could have analysed different sets of projects and therefore come to different conclusions. Additionally, the timeframe of a study can impact its results, as technology is constantly changing. Another concern could be publication bias, in which certain perspectives or outcomes are disproportionately represented in the literature, potentially distorting the overall understanding of a particular topic.

6 Conclusion

To summarise, examining design patterns in microservices architecture has yielded useful insights for practitioners. The adoption of MSA patterns is influenced by various contextual factors, such as project requirements, current technology, and team dynamics. Well-known patterns, including API Gateway, circuit breaker, load balancer, service registry, and health check, have been identified, with possible variations in ranking due to project specificities. The prioritisation of quality attributes such as scalability, flexibility, testability, performance, and elasticity highlights the importance of considering these factors when selecting design patterns for success in MSA. Continued research in this area can benefit from exploring other contextual factors that may affect decision-making. Ultimately, bridging the gap between academia and industry practices is vital for advancing the understanding and implementation of MSA design patterns. In future work, implementing these identified patterns in real-life project scenarios and measuring their impact can further validate or challenge the findings presented.

References

[1] Amazon Web Services. (n.d.). Microservices on AWS. Retrieved from https://aws.amazon.com/microservices

[2] Rodgers, Peter Service-Oriented development on NetKernel- Patterns, Processes & products to reduce system complexity. (n.d.). In *Web Services Edge 2005 East: CS-3*. https://web.archive.org/web/20180520124343/http://www.cloudcomputingexpo.com/node/80883

[3] Bogner, J., Wagner, S., & Zimmermann, A. (2019). Using architectural modifiability tactics to examine evolution qualities of Service- and Microservice-Based Systems. Sics Software-intensive Cyber-physical Systems, 34(2–3), 141–149. https://doi.org/10.1007/s00450-019-00402-z

[4] Waseem, M., Liang, P., Shahin, M., Di Salle, A., & Márquez, G. (2021). Design, monitoring, and testing of microservices systems: The practitioners' perspective. Journal of Systems and Software, 182, 111061. https://doi.org/10.1016/j.jss.2021.111061

[5] G. Márquez and H. Astudillo, "Actual Use of Architectural Patterns in Microservices-Based Open Source Projects," 2018 25th Asia-Pacific Software Engineering Conference (APSEC), Nara, Japan, 2018, pp. 31-40, doi: 10.1109/APSEC.2018.00017.

[6] Márquez, G., & Astudillo, H. (2018). Actual Use of Architectural Patterns in Microservices-Based Open Source Projects. . https://doi.org/10.1109/apsec.2018.00017

[7] Esas, Ömer. (2021-2022). Design Patterns and Anti-Patterns in Microservices Architecture: A Classification Proposal and Study on Open Source Projects. Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica. Advisor: Prof. Elisabetta Di Nitto.

[8] Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. Journal of Systems and Software, 150, 77–97. https://doi.org/10.1016/j.jss.2019.01.001

[9] Waseem, M., Liang, P., Ahmad, A., Shahin, M., Khan, A. A., & Márquez, G. (2022). Decision Models for Selecting Patterns and Strategies in Microservices Systems and their Evaluation by Practitioners. . https://doi.org/10.1109/icse-seip55303.2022.9793911

[10] Waseem, M., Liang, P., Shahin, M., Di Salle, A., & Márquez, G. (2021b). Design, monitoring, and testing of microservices systems: The practitioners' perspective. Journal of Systems and Software, 182, 111061. https://doi.org/10.1016/j.jss.2021.111061

[11] Aksakalli, I. K., Çelik, T., Can, A. B., & Teküerdoğan, B. (2021). Deployment and communication patterns in microservice architectures: A systematic literature review. Journal of Systems and Software, 180, 111014.

[12] Smela B, Toumi M, Świerk K, Francois C, Biernikiewicz M, Clay E, Boyer L. Rapid literature review: definition and methodology. J Mark Access Health Policy. 2023 Jul 28;11(1):2241234. doi: 10.1080/20016689.2023.2241234. PMID: 37533549; PMCID: PMC10392303.

[13] Alashqar, A. M., & Kurdya, Z. (2022). Examining the Design Patterns of Microservices for Achieving Performance Quality Tactics. International Journal of Academic Information Systems Research (IJAISR), 6(12), 4-13.

[14] Hubian. (2022, July 5). Microservice Architecture and Design Patterns.Medium.https://medium.com/@hubian/microservice-architecture-and-design-patterns-ada1f9523ed7

[15] Osses, F., Márquez, G., & Astudillo, H. (2018). Exploration of academic and industrial evidence about architectural tactics and patterns in microservices. . https://doi.org/10.1145/3183440.3194958

[16] Bass, L., Clements, P., & Kazman, R. (2013). Software Architecture in Practice (3rd Edition). SEI Series in Software Engineering.

[17] Team, I. C. (2023, November 23). *SOA vs. Microservices: What's the Difference? - IBM Blog*. IBM Blog. https://www.ibm.com/blog/soa-vs-microservices/

[18] Team, C. A. (2023, July 27). Advantages and Disadvantages of Microservices Architecture. *Cloud Academy*. https://cloudacademy.com/blog/microservices-architecture-challenge-advantage-drawback/

[19] Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Сафина, Л. (2017). Microservices: yesterday, today, and tomorrow. In *Springer eBooks* (pp. 195–216). https://doi.org/10.1007/978-3-319-67425-4_12

[20] Zhang, H., Li, S., Jia, Z., Zhong, C., & Zhang, C. (2019). Microservice Architecture in Reality: An Industrial Inquiry. *2019 IEEE International Conference on Software Architecture (ICSA)*. https://doi.org/10.1109/icsa.2019.00014

[21] Lu, Z., Delaney, D., & Lillis, D. (2023). A survey on Microservices Trust Models for Open Systems. *IEEE Access*, *11*, 28840–28855. https://doi.org/10.1109/access.2023.3260147

[22] Söylemez, M., Tekinerdoğan, B., & Tarhan, A. K. K. (2022). Challenges and Solution Directions of Microservice Architectures: A Systematic Literature review. *Applied Sciences*, *12*(11), 5507. https://doi.org/10.3390/app12115507

[23] Villamizar, M., Garcés, O., Castro, H., Verano, M., Salamanca, L., Casallas, R., & Gil, S. (2015). Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud. . https://doi.org/10.1109/columbiancc.2015.7333476

[24] Sampaio, A. R., Kadiyala, H., Hu, B., Steinbacher, J., Erwin, T., Rosa, N., Beschastnikh, I., & Rubin, J. (2017). Supporting Microservice Evolution. *IEEE International Conference on Software Maintenance and Evolution*. https://doi.org/10.1109/icsme.2017.63

[25] Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Casallas, R., Gil, S., Valencia, C., Zambrano, A., & Lang, M. (2017). Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures. *Service Oriented Computing and Applications*, 233–247. https://doi.org/10.1007/s11761-017-0208-y

[26] Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Casallas, R., Gil, S., Valencia, C., Zambrano, A., & Lang, M. (2016). Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures. *IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing*. https://doi.org/10.1109/ccgrid.2016.37

[27] Singh, A., Singh, V., Aggarwal, A., & Aggarwal, S. (2022). Event Driven Architecture for Message Streaming data driven Microservices systems residing in distributed version control system. *2022 International Conference on Innovations in Science and Technology for Sustainable Development (ICISTSD)*. https://doi.org/10.1109/icistsd55159.2022.10010390

[28] Li, C. (2005). Real-time event driven architecture for activity monitoring and early warning. *Emerging Information Technology Conference, EITC*. https://doi.org/10.1109/eitc.2005.1544382

[29] Almeida, R. B., Junes, V. R. C., Machado, R., Da Rosa, D. Y. L., Donato, L. M., Yamin, A., & Pernas, A. M. (2019). A distributed event-driven architectural model based on situational awareness applied on internet of things. *Information & Software Technology*, *111*, 144–158. https://doi.org/10.1016/j.infsof.2019.04.001

[30] Levina, O., & Stantchev, V. (2009). Realizing Event-Driven SOA. *International Conference on Internet and Web Applications and Services (ICIW)*. https://doi.org/10.1109/iciw.2009.14

[31] Ghalsasi, S. Y. (2009). Critical success factors for event driven service oriented architecture. *ICIS '09: Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*. https://doi.org/10.1145/1655925.1656191

[32] Rahmatulloh, A., Nugraha, F., Gunawan, R., & Darmawan, I. (2022). Event-Driven Architecture to Improve Performance and Scalability in Microservices-Based Systems. *International Conference on Advancement in Data Science, E-learning and Information Systems (ICADEIS)*. https://doi.org/10.1109/icadeis56544.2022.10037390

[33] Wei, T., Coady, Y., MacDonald, J. A., Booth, K. S., Salter, J., & Girling, C. (2017). Dumb pipes for smart systems: How tomorrow's applications can salvage yesterday's plumbing. *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. https://doi.org/10.1109/pacrim.2017.8121899

[34] Buchgeher, G., Winterer, M., Weinreich, R., Luger, J., Wingelhofer, R., & Aistleitner, M. (2017). Microservices in a small development organization. In *Lecture Notes in Computer Science* (pp. 208–215). https://doi.org/10.1007/978-3-319-65831-5_15

[35] Poster: Exploration of Academic and Industrial Evidence about Architectural Tactics and Patterns in Microservices, Felipe Osses, Gastón Márquez, H. Astudillo, 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion

[36] Understanding Quality Attributes in Microservice Architecture, Shanshan Li, 2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW)

[37] Understanding and addressing quality attributes of microservices architecture: A Systematic literature review, Shanshan Li, He Zhang, Zijia Jia, Chenxing Zhong, Cheng Zhang, Zhihao Shan, Jinfeng Shen, Muhammad Ali Babar, Inf. Softw. Technol.

[38] Burns, B., & Oppenheimer, D. (2016). *Design patterns for container-based distributed systems*. USENIX. https://www.usenix.org/conference/hotcloud16/workshop-program/presentation/burns

[39] Montesi, F., & Weber, J. (2016). *Circuit breakers, discovery, and API gateways in microservices*. https://www.semanticscholar.org/paper/Circuit-Breakers%2C-Discovery%2C-and-API-Gateways-in-Montesi-Weber/00025f8caf92b2e737027790b36c82208d9c1ac7

[40] Aversano, L., Canfora, G., Cerulo, L., Del Grosso, C., & Di Penta, M. (2007). An empirical study on the evolution of design patterns. *ESEC-FSE '07*. https://doi.org/10.1145/1287624.1287680

[41] Zhang, Yanqi & Gan, Yu & Delimitrou, Christina. (2019). uqSim: Scalable and Validated Simulation of Cloud Microservices.

[42] Rocha, F., Soares, M., & Rodriguez, G. (2023). Patterns in Microservices-based Development: A Grey Literature Review. In *Anais do XXVI Congresso Ibero-Americano em Engenharia de Software*, (pp. 61-76). Porto Alegre: SBC. doi:10.5753/cibse.2023.24693

[43] Garousi, V., Felderer, M., & Mäntylä, M. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information & Software Technology*, *106*, 101–121. https://doi.org/10.1016/j.infsof.2018.09.006

[44] Taibi, D., Lenarduzzi, V., & Pahl, C. (2018). Architectural Patterns for Microservices: A Systematic Mapping Study. *Proceedings of the 8th International Conference on Cloud Computing and Services Science CLOSER - Volume 1*. https://doi.org/10.5220/0006798302210232

[45] McLellan, H. (2020). Editorial: The importance of Peer review. *Canadian Journal of Emergency Nursing*. https://doi.org/10.29173/cjen45

[46] Wendorff, P. (2002). Assessment of design patterns during software reengineering: lessons learned from a large commercial project. *Fifth European Conference on Software Maintenance and Reengineering*. https://doi.org/10.1109/.2001.914971

[47] Dong, J., Zhao, Y., & Tu, P. (2009). A REVIEW OF DESIGN PATTERN MINING TECHNIQUES. *International Journal of Software Engineering and Knowledge Engineering*, *19*(06), 823–855. https://doi.org/10.1142/s021819400900443x

[48] Chambers, C., Harrison, B., & Vlissides, J. (2000). A debate on language and tool support for design patterns. *ACM-SIGACT Symposium on Principles of Programming Languages*. https://doi.org/10.1145/325694.325731

[49] Cross, N., & Cross, A. (1995). Observations of teamwork and social processes in design. *Design Studies*, *16*(2), 143–170. https://doi.org/10.1016/0142-694x(94)00007-z

[50] Osses, F., Márquez, G., & Astudillo, H. (2018b). Exploration of academic and industrial evidence about architectural tactics and patterns in microservices. *ICSE '18: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*. https://doi.org/10.1145/3183440.3194958

[51] Riesen, R. (2006). Communication patterns [message-passing patterns]. *Proceedings 20th IEEE International Parallel & Distributed Processing Symposium*. https://doi.org/10.1109/ipdps.2006.1639567

[52] Forsythe, C., Joseph, N., Szajnfarber, Z., & Gralla, E. (2019). Visualizing communication patterns in design teams over time. In *Springer eBooks* (pp. 449–461). https://doi.org/10.1007/978-3-030-00114-8_37

[53] Larson, S. (n.d.). *Building an API Gateway with NGINX*. Grizzly Peak Software. https://www.grizzlypeaksoftware.com/articles?id=4vlfDp2ZanpAh3bSuUzPeZ

[54] Chibuike, N. (2024, January 24). Backends for Frontends — my take - FAUN — Developer Community. https://faun.pub/backends-for-frontends-my-take-b97663ec1b68

Microservices Architectural Patterns and Quality Requirements: A Rapid Literature Review

Elena Georgiou s5772036

Abstract—Microservices architecture has gained significant prominence in modern software development due to its scalability, improved fault isolation, and reusability across different systems. Numerous studies have explored various patterns of microservices and their impact on quality attributes. However, there is a need to consolidate and synthesize the findings from these studies to obtain a more holistic understanding of this relationship. In this study, we extend the latest literature review in the field to understand the current state-of-the-art in the area. We conduct, a rapid literature review on white literature published between the years 2019-2023. Our results show that the identified gaps of the last review have not been filled yet and identify more architectural patterns and quality attributes that require further investigation.

Index Terms—Software architecture, microservices, architectural patterns, quality attributes.

1 Introduction

In the face of the growing complexity of computer software and the unwavering desire for agility, the constraints of monolithic architectures are becoming more visible. As a response, microservice architecture has arisen in recent years, offering multiple benefits, including shorter software delivery cycles, more efficient maintenance, and scalability [13].

The task of dividing monolithic applications into microservices is complex and architectural patterns step into the game to provide applicable solutions to common problems of software architecture. However, selecting the best patterns for a particular project can be challenging due to the vast array of available options.

Over the past decade, numerous research studies have been conducted to investigate the influence of different microservices architectural patterns on quality attributes. To monitor the increasing number of new works in this field, several literature reviews have also been conducted during the past years, aiming to provide a comprehensive overview of the state-of-the-art. These reviews aim to offer a comprehensive overview of the current state-of-the-art. Based on a thorough examination of existing literature, and to the best of our understanding, the most recent literature review [17], was published in 2021 and focused on examining the correlation between patterns and quality attributes based on literature up until 2019.

This study aims to extend the latest review in the field, to identify the current state-of-the-art between the period 2020-2023. We are conducting a rapid literature review on academic literature published in the aforementioned period.

More specifically, we aim to answer the following research question:

RQ: How are quality attributes influenced as a result of applying different microservices patterns?

With the above research question, our objective is to understand how the research focus has shifted during the past four years regarding the influence of microservices patterns on quality attributes.

With this, we aim to assist the practitioners in their decision-making, as well as identifying possible research gaps that exist in the area. Furthermore, we aim to understand whether the research gaps that were identified by the previous work have been filled.

This document is divided into the following sections: Section 2 defines the main concepts and terms that are used in the paper. Section 3 presents the related work, while Section 4 explains the study design. Section 5 presents the results of the Research Question, which are discussed in Section 6. In Section 7, we mention potential threats to the validity of this paper, while in Section 8 we form our conclusion.

• *Elena Georgiou is with the University of Groningen, E-mail: e.georgiou.5@student.rug.nl.*

2 Background

This section briefly introduces the concept of microservice architecture and provides clear definitions of the foundamental terms that are used throughout this paper.

2.1 Microservices architecture

The foundation of microservices emerged from Service-Oriented Architecture, which was first introduced in the late 1990s and early 2000s. Microservices decompose traditional monolithic applications into a collection of services that are independently deployable and loosely coupled [13]. They are lightweight by nature and collaborate with similar services through well-defined interfaces. They communicate using lightweight protocols like asynchronous message buses [10]. Microservices belonging to the same application can be developed using different frameworks, programming languages, and resources. This approach enhances application scalability, performance, maintainability, and interoperability [8].

2.2 Microservices architectural patterns

Dividing monolithic applications into microservices is a complex task. Microservices architectural patterns [16] provide widely applicable solutions to recurring problems in software development, defining the structure and interaction between microservices. These patterns address issues such as application decomposition, data management, and services communication.

Numerous patterns have emerged for microservices architecture, covering various implementation aspects. Each pattern is designed to address specific system architecture problems and requirements. Selecting and implementing a pattern can positively affect certain quality attributes while negatively affecting others.

2.3 Quality Attributes

A Quality Attribute (QA), as defined by Bass et al. [3], is a ”measurable or testable property of a system that is used to indicate how well the system satisfies the needs of its stakeholders”. The architecture design of software systems is the initial stage of software development in which quality attributes can be met.

3 Related Work

Empirical studies have shown that identifying architectural patterns that meet certain quality attributes is among the most challenging aspects of software architecture design [7].

A study conducted in 2018 [12] gathered a total of 80 architectural patterns of microservices from academia and 44 from industry. These

Table 1. Microservices Keywords and related terms used in the search query

Keyword	Related terms
Microservice*	micro-service*, micro service*, architecture, service-oriented architecture, SOA
Architectural pattern*	architectural style*, architectural tactic*, microservice* pattern
Quality attribute	attribute*, quality, architectural driver, QA, requirement*

patterns were identified and analyzed through an examination of 1067 studies resulting in a selection of 69 studies published between the years 2015-2017. Based on this analysis, they proposed a taxonomy to categorize these patterns based on their context.

Another study conducted in 2019 [18] ties together microservices patterns and quality attributes. They do this, by performing a multivocal literature review from white and grey literature. They identified 54 patterns in total, which they grouped into six categories based on their benefits: data persistence, communication, entry-point, distribution, fault tolerance, and supplements. They focus on 6 quality attributes, that they found the most related to the identified patterns based on the literature. These 6 attributes ordered in descending order based on frequency are: maintainability, reliability, security, performance efficiency, compatibility, and portability.

To the best of our knowledge, Valdivia et al. [17] conducted the most recent study that attempted to identify the impact of architectural patterns on quality attributes. The study was conducted in 2021 and they analysed the literature of the period 2014-2019. One of their most important findings was the research gap in the association of two quality attributes, suitability and usability, with microservices architectural patterns. No patterns were found to meet these two attributes.

4 Study Design

In this section, we explain in detail the study design, including the research questions and aims, the methodology that was followed to find relevant literature, and the extraction and synthesis procedure.

4.1 Research Aims and Questions

In this study, we conduct a rapid review to explore the connection between patterns and quality attributes. More specifically, the goal of this study is to answer the following research question (RQ):

RQ: How are quality attributes influenced as a result of applying different microservices patterns?

With this research question at the forefront, we have set two overarching objectives. The first objective aligns with the primary goal of rapid reviews [4], which is to support practitioners in their decision-making on microservices architecture design. Equally significant is our second objective, which involves identifying gaps in existing research on microservices architectural patterns and initiating further primary studies to address these gaps. While this may not fall within the typical use cases of rapid reviews, our study serves as an indicator of potential gaps that can be later verified and, if necessary, filled.

Our research is centred around analyzing literature published from 2020 onwards. This time frame was chosen as the most recent secondary study in the field was conducted using data up until 2019 ([17]), as explained in Section 3. To our knowledge, there have been no published studies since 2020 that investigates this relationship. Thus, this motivates us to perform this study.

4.2 Review Protocol

The review protocol is based on guidelines recommended by B. Cartaxo et al. ([4], which provide a comprehensive guide for software engineering researchers conducting rapid reviews.

4.2.1 Search strategy

To abbreviate the search for research papers, and conduct the rapid review under the agreed time frame, we used a single search engine, namely *Scopus*. This source was selected, as it is recommended in the guidelines ([4]) due to its wide coverage of research papers from major digital libraries.

For the automated search, we identified keywords used in previous microservices reviews ([17], [12], [21]) and supplemented them with related concepts, as shown in Table 1. In our query, we also incorporated the asterisk wildcard character (*), which can be replaced by zero or more characters [1]. The keywords were used to search for matches in the title, abstract, and index terms (keywords) of the papers within the search engine.

4.2.2 Search Criteria

In order to limit the results, inclusion and exclusion criteria were applied. The inclusion criteria are: (1) The study must be in the context of Computer Science or Engineering. (2) The study must be published after 2019. The start date was selected based on that the most recent secondary study known to us in the field. This study ([17]) was conducted using data up until 2019, as explained in Section 3. (3) The study must be written in English. (4) The study must provide answers to our RQ. (5) The study must be arbitrated. To enhance the quality of this study, the studies must have been accepted to be published by third parties (peer-reviewed). A single exclusion criterion was included; The study is not a primary study.

By integrating the above inclusion and exclusion criteria into the search string, we refined the results to focus on the most potentially relevant ones. The resulting query is as follows:

```
TITLE-ABS-KEY (
("Microservice*" OR "Micro-service*" OR "Micro service*" OR
"architecture" OR "Service-oriented Architecture" OR "SOA")
AND ("Architectural Pattern*" OR "architectural style*" OR "architectural tactic*" OR "microservice pattern")
AND ("attribute*" OR "quality" OR "driver*" OR "QA" OR "Requirement*")
) AND (LIMIT-TO (SUBJAREA,"COMP") OR LIMIT-TO (SUBJAREA,"ENGI"))
AND PUBYEAR > 2019 AND PUBYEAR < 2025
AND (LIMIT-TO (LANGUAGE,"English"))
AND (LIMIT-TO (SRCTYPE,"j") OR LIMIT-TO (SRCTYPE,"p"))
AND (EXCLUDE (DOCTYPE,"re"))
```

4.2.3 Selection Procedure

Our methodology for determining the relevance of the papers involved a multi-round process. All rounds were performed by the same person, i.e., the author. In every round, papers that did not meet one or more of the criteria were discarded.

The search yielded a total of 155 papers. In the first round, the titles of the studies were examined. To minimize the risk of false negatives, a lenient approach was adopted during the discarding process. In this round, 79 papers were excluded from further analysis. In the second round, the abstract, introduction, and conclusion of the papers were analysed. In this round, we maintained flexibility by considering papers that didn't immediately demonstrate a direct connection with our research question but showed promise in providing relevant information. In this round, 67 papers were discarded. The papers that showed potential relevance from these sections, continued to the third and final round, where we went through the whole paper.

The outcome of each round of the selection process, along with explanations for the exclusion of specific papers, can be accessed online.[1]. The final list consists of 6 papers, which are shown in Table 2.

4.3 Extraction, Analysis and Synthesis

In this step, we extracted all relevant findings that could help to answer our research questions. The extraction was also conducted by a solo

[1]Results of selection procedure: `https://shorturl.at/bpLS4`

[H]

Table 2. Relevant Publications (in ascending alphabetical order)

Title	Publication year	Format
A Method for Architectural Trade-off Analysis Based on Patterns: Evaluating Microservices Structural Attributes [6]	2020	Conference Paper
Architectural strategies for interoperability of software-intensive systems: Practitioners' perspective [20]	2021	Conference Paper
Cheetah: A High-Speed Programmable Load-Balancer Framework with Guaranteed Per-Connection-Consistency [2]	2022	Journal Article
Designing Microservice Systems Using Patterns: An Empirical Study on Quality Trade-Offs [19]	2022	Conference Paper
How to design Future-Ready Microservices? Analyzing microservice patterns for Adaptability [5]	2023	Journal Article
Microservices Centric Architectural Model for Handling Data Stream Oriented Applications [14]	2020	Journal Article

researcher, as was the selection procedure.

We performed a narrative synthesis [4], as we used words and text to summarize our findings. In order to have an overview of all collected data, we incorporated them in a table. As we are investigating the relationship between two fields (patterns and quality attributes), a table is ideal. The rows corresponded to each of the reported patterns found, while each column corresponded to the quality attributes.

The next step was to report the findings of this study. We decided to include in the tables that are discussed in Section 5.3 (Table 3, Table 4) solely those patterns and quality attributes that do not contribute to a research gap, allowing for efficient use of space. Those that are related to a research gap are discussed in Section 5.4. The complete table is available for online access [2].

5 Results

In this section, we present the results of our study by answering our research questions. More specifically, in the first section, we give a brief overview of the selected publications. In the second subsection, we give a precise definition of the quality attributes, which are used to describe the impact of the microservices patterns in the third subsection.

5.1 Publications selected

Table 2 lists all the selected publications. In this section, we provide a concise overview of each selected publication.

A study by Vale et al. [19] aims to investigate the impact of software patterns on quality attributes based on practitioners' perceptions. They conducted nine semi-structured interviews with industry professionals, focusing on the professionals' knowledge and adoption of software patterns, the perceived architectural trade-offs associated with the patterns, and the metrics used by professionals to measure quality attributes. By gathering insights from these interviews, the study aimed to validate and enhance the understanding of the impact of 14 specific patterns on 7 quality attributes.

Rosa et al. [6] presented a systematic method for analyzing architectural trade-offs in microservices patterns, intending to guide software architects in identifying the most appropriate architectural patterns for a given context. One valuable contribution of this study is the evaluation of 24 architectural patterns based on their impact on three quality attributes: coupling level, module/service size, database sharing between modules/services.

Daniel et al.[5] analyzed how microservices patterns affect the extensibility of microservice-based systems, based on their context and solution. More specifically, they analysed the impact of 18 architectural patterns on three dimensions of extensibility: Microservice Internal Flexibility (adaptability), Microservice Extensibility, and System Extensibility.

In a work by Valle et al. [20], the focus was on examining how various architectural strategies impact the interoperability of software. To gather insights, an online survey was conducted, involving experienced practitioners with more than five years of industry experience. Among others, the study aimed to understand the practitioners' perspectives on the influence of different microservices architectural patterns, such as pipes and filters and shared databases, on various quality attributes.

Philip et al.[14] worked on the integration of a combinational architectural design to realize any data stream-oriented application. They achieved this, by using microservices as a central component, complemented by the pipes and filter and Model-View-Controller (MVC) architectural patterns. Despite this paper not directly addressing our research question regarding the relationship between architectural patterns and quality attributes, we deemed it important to include it in our results. This is because it generates a contradictory finding, which will be discussed in the subsequent sections of this paper.

A study by Barbette et al.[2] leveraged the benefits of load balancers within microservices to develop Cheetah, a load-balancer framework. This study was used in our paper to assess the advantages and disadvantages of utilizing an enhanced version of Load Balancers that surpass the capabilities of the conventional ones. Although the literature does not unanimously agree on whether load balancers are considered microservices patterns, we chose to integrate this approach, as it is a solution that can be widely used in microservices due to its advantages.

5.2 Quality attributes

In the following paragraphs, we will provide brief definitions of the quality attributes that have been addressed during the examined period in the microservices area. These attributes will be used to address the Research Question.

Availability refers to the extent to which a system is accessible and operational when needed, with minimal downtime or disruptions [3]. It is used interchangeably with reliability [9].

Elasticity of a system refers to its ability to dynamically adjust its resources and capacity, by scaling up or down, in response to changing workload or demand [11]. Despite being mentioned only a few times in the literature, it is essential to give careful attention to this quality attribute as it is closely associated with scalability [12].

Maintainability refers to how easily a software system can be modified, updated, repaired, or extended over time [3]. It is a balancing act between the time and resources required to apply the new changes.

Microservice Internal Adaptability pertains to the capability of enhancing the functionality of a service while keeping its interface unchanged [5]. While microservice internal adaptability contributes to microservice maintainability, they are not interchangeable terms. Internal adaptability focuses specifically on the ability of a microservice to evolve internally without changes in the interface, while maintainability encompasses a broader range of characteristics related to the ease of maintaining the microservice throughout its lifecycle.

Microservice External Extensibility is defined as the capability of a single microservice to integrate and accommodate new features or functionalities [5].

Monitorability refers to a system's capability to be effectively monitored and observed at runtime for various purposes, such as performance analysis, troubleshooting, and identifying anomalies [3]. Monitorability can also be found in the literature as observability [19].

Performance is a metric that assesses a system's capacity to fulfil time constraints while handling inputs, requests, or events [3].

[2]Extracted data: `https://shorturl.at/tAJP3`

Table 3. Relationship of reported patterns with quality attributes (part1)

Pattern	Availability	External Extensibility	Internal Adaptability	Maintainability	Monitorability
Ambassador	[19]	[5]	[5]	[19]	[19]
Backends for frontends		[5]	[5]	[19]	
Choreography		[5]	[5]		
Claim check		- [5]	- [5]		
Command-side replica		[5]	[5]		
CQRS	[19]	[5]	[5]		
Database per service		[5]	- [5]		
Decompose by business capabilities		[5]	[5]		
Decompose by subdomain		[5]	[5]		
Event sourcing			- [5]		
External Configuration Store	[19]			[19]	
Gatekeeper		[5]	[5]		
Gateway aggregation					
Gateway Offloading				[19]	[19]
Gateway routing			[5]		
Load Balancer	[2]				
Messaging		[5]		[20]	
Pipes and filters	(- [14])	[5]	[5]	[19](, [14])	- [19], [20]
Queue-based load leveling			[5]		
Shared database		- [5]	- [5]		
Sidecar		[5]	[5]	[19]	
Static Content Hosting	[19]			[19]	
Strangler				[19]	

Scalability is the ability of a system to handle increasing demands or a growing workload (high volumes of data, users, transactions, etc.) effectively without sacrificing performance or requiring significant changes to its infrastructure [3].

Security corresponds to a system's ability to protect data from unauthorized access while allowing authorized users and systems to access it [3].

System extensibility, on the other hand, is defined as the capacity of the overall system to incorporate and integrate new microservices into its architecture [5].

5.3 Relationship of microservices patterns and QA as per existing literature

Table 3 and Table 4 provide a summary of the architectural patterns that have been supported by evidence in the literature in terms of the quality attributes they fulfil as well as those they do not. The two tables illustrate the relationship of the identified patterns with two different sets of quality attributes. In case there is evidence that a certain pattern does not meet a quality attribute, it is indicated by "-" followed by the reference to the corresponding publication.

As it can be observed, the pattern that received the highest level of support from the literature during the years 2020-2023 is Pipes and Filters, as it was found in 4 out of the 6 sources that were used to answer our RQ.

Ambassador and *Pipes and Filters*, were among the patterns that were reported to have the most quality attributes based on the analysed literature. However, the latter received contradictory evidence regarding its quality attributes, as it was identified as simultaneously meeting and not meeting a quality attribute based on different sources. More specifically, conflict evidence was found regarding its *Monitorability, Performance,* and *Scalability*. One potential reason regarding the former two QAs is that one of the studies was based on the practitioners' perspective. Thus, it may be the case that their opinion is different from reality.

The quality attributes that can be found in most of the patterns are *External Extensibility* and *Internal Adaptability*. Also, scalability is the most popular quality attribute, as it was examined by most of the found sources.

5.4 Identified research gaps

An important remark is that many patterns and QAs lacked substantial support in the literature of the examined period. These cases are presented here. The exact relationship that was identified regarding the below patterns and QAs is not reported in the tables discussed above due to spatial constraints but can be accessed online [3].

The quality attributes that had limited evidence, with support by only one source and relationship with two or fewer patterns, include *Portability* [20], *Flexibility* [14], *Testability* [19], *Usability* [20], *Reliability* [20], and *Functionality* [20]. Specifically, the first three were associated with only one pattern according to the literature, while the latter three were linked to two patterns.

Furthermore, numerous patterns were supported in the literature by only one source. These patterns are: *Adapter Microservice, Anti-corruption layer, API Composition, Application Metrics, Asynchronous request-reply, Business Microservice, Change Code dependency to service call, Compare Resource Consolidation, Decompose the Monolith Based on Data Ownership, Domain Driven, Introduce Edge Server, Log Aggregation, Leader Election, Self-Contained Server, SAGA, Scalable Store, Service Registry, Transactional Outbox*.

6 Discussion

In this section, we discuss the results by providing some important remarks about them.

As it was observed, *External Extensibility* and *Internal Adaptability* were found to be the quality attributes that most patterns meet. This is not necessarily representative of reality, as there was a specific study about the impact of several patterns on these two quality attributes. Thus, this just implies that there is more evidence supporting this particular attribute. An interesting remark is that within the same study, the evaluation of *System Extensibility* was conducted on the same patterns. However, it was observed that a significant number of patterns did not meet this quality attribute, indicating a limited positive impact of patterns on system extensibility.

Furthermore, in the results, it is noted that the scalability had the highest level of attention in the literature. This outcome aligns with

[3] https://shorturl.at/tAJP3

Table 4. Relationship of reported patterns with quality attributes (part2)

Pattern	Performance	Scalability	Security	System Extensibility	Additional benefits and drawbacks
Ambassador		[19]	[19]		- Increases service coupling level [6]
Backends for frontends	[19]				
Choreography				[5]	
Claim check				- [5]	
Command-side replica					- Increases service size [6]
CQRS	[19]	[19]		[5]	- Increases number of database sharing between services [6] Decreases service coupling level [6]
Database per service				- [5]	Decreases number of database sharing between services [6] Decreases service coupling level [6]
Decompose by business capabilities					Decreases service size [6] Decreases service coupling level [6]
Decompose by subdomain					Decreases service size [6] Decreases service coupling level [6]
Event sourcing				[5]	Decreases number of database sharing between services [6] Decreases service coupling level [6]
External Configuration Store		[19]	[19]		
Gatekeeper					
Gateway aggregation	[19]	- [6], - [19]			- Single point of failure [6]
Gateway Offloading	[19]		[19]		
Gateway routing				[5]	
Load Balancer		[2]			
Messaging					Decreases number of database sharing between services [6] Decreases service coupling level [6]
Pipes and filters	[19], - [20]	[19], (-[14])	-[20]	[5]	
Queue-based load leveling					
Shared database	[20]		[20]	- [5]	- Increases service coupling level [6] - Increases number of database sharing between services [6]
Sidecar		[19]	[19]	[5]	- Increases service coupling level [6]
Static Content Hosting	[19]		[19]		
Strangler		[19]			

expectations since scalability aligns closely with the primary objective of microservices architecture. Given the emphasis on building systems that can dynamically adjust and accommodate varying workloads, it is not surprising that scalability emerged as a focal point in the literature and research surrounding microservices.

An interesting finding is that, despite the previous review - conducted on literature up to 2019 - ([17]) highlighting a research gap related to patterns that fulfil usability and functional suitability (functionality), this gap persisted during the 2020-2023 timeframe examined in the current study. This indicates a need for future work to explore potential patterns that address these attributes.

Furthermore, a plausible explanation for the gaps observed in the literature on a large number of patterns could be that they were extensively studied and explored in previous years. It is possible that these patterns underwent thorough examination and analysis during earlier research efforts, resulting in relatively fewer recent publications or discussions on them.

7 Threats to validity

In this section, we address potential factors that could impact the accuracy and dependability of our paper. The main concerns arise from the fact that we conducted a Rapid Review due to time constraints.

The primary issue regarding the validity of our study pertains to the size of our sample and the search strategy outlined in Section 4. The time limitations of the rapid review may have restricted the inclusion of a comprehensive range of papers, potentially resulting in the exclusion of relevant ones. A larger pool of papers would have provided a more diverse and comprehensive understanding of the current literature on architectural patterns.

Another significant point to mention is that the quality appraisal step was omitted, which involves assessing the selected papers' quality in pairs. The extraction procedure was conducted by a single reviewer as well, introducing the possibility of bias.

Another limitation of this study, attributed to its nature as a rapid review, concerns the synthesis procedure employed. We utilized a narrative synthesis approach, following the guidelines[4]. This lightweight method offers a less time-consuming and effort-intensive approach to knowledge synthesis, but it also raises concerns about potential threats to validity.

8 Conclusion and Future Work

This paper conducts a rapid literature review, regarding the state-of-the-art in microservices systems. More specifically, it focuses on identifying the impact of various patterns on quality attributes.

The findings show that the most strongly supported architectural pattern is *Pipes and Filter*. This was also found to be the pattern that meets the most quality attributes, along with *Ambassador*.

A key finding of this study is that a gap that was identified by a previous review still exists in the study. The research gap is about the lack of support regarding the patterns that meet two quality attributes; usability and functional suitability. This indicates that future work is needed to identify the proposed patterns in the microservices research area to meet these attributes.

The research gaps we identified are indicative of the current state of the art in microservices, as this is a rapid review. As a future work, one could query also Google Scholar, as it is another search engine that is recommended in the guidelines used [4]. Moreover, for a more systematic approach, more search engines and databases could be queried with the aim of identifying more papers and getting a more holistic overview of the current state-of-the-art in this area. The search query could also be updated with more related concepts that can be found in the literature. For instance, the term *non-functional requirements*,

which is usually abbreviated as *NFR*, is another widely used term that can be found in the literature instead of quality attributes. We were not aware of this term before this study, so it was not included in the search string.

Last but not least, another aspect that we would like to bring to attention, is that despite diligently following a systematic approach per the recommended guidelines, and thoroughly reviewing over 150 papers, we were able to identify only six papers that were relevant to our aim. Throughout the process of searching for relevant papers, we encountered numerous articles(e.g., [15]) that focused on the implementation of automated systems designed to migrate monolithic applications to microservices. This observation suggests that the scientific community may have shifted its attention towards the development of more automated solutions aimed at facilitating the development of microservices.

REFERENCES

[1] Scopus search guide. `https://schema.elsevier.com/dtds/document/bkapi/search/SCOPUSSearchTips.htm`. Accessed: 2024-03-20.

[2] T. Barbette, E. Wu, D. Kostić, G. Q. Maguire, P. Papadimitratos, and M. Chiesa. Cheetah: A high-speed programmable load-balancer framework with guaranteed per-connection-consistency. *IEEE/ACM Transactions on Networking*, 30(1):354–367, 2022.

[3] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice (SEI Series in Software Engineering)*. Addison-Wesley Professional, third edition, 2012.

[4] B. Cartaxo, G. Pinto, and S. Soares. *Contemporary Empirical Methods in Software Engineering*, chapter 13, pages 357–384. Springer, 2020.

[5] J. a. Daniel, X. Wang, and E. Guerra. How to design future-ready microservices? analyzing microservice patterns for adaptability. In *Proceedings of the 28th European Conference on Pattern Languages of Programs*, EuroPLoP '23, New York, NY, USA, 2024. Association for Computing Machinery.

[6] T. de Oliveira Rosa, J. a. F. L. Daniel, E. M. Guerra, and A. Goldman. A method for architectural trade-off analysis based on patterns: Evaluating microservices structural attributes. In *EuroPLoP '20: Proceedings of the European Conference on Pattern Languages of Programs 2020*, pages 1–8, July 2020.

[7] G. Huang, H. Mei, and F.-Q. Yang. Runtime recovery and manipulation of software architecture of component-based systems. *Automated Software Engineering*, 13:257–281, 04 2006.

[8] G. Kecskemeti, A. C. Marosi, and A. Kertesz. The ENTICE approach to decompose monolithic services into microservices. In *2016 International Conference on High Performance Computing Simulation (HPCS)*, pages 591–596, 2016.

[9] S. Li, H. Zhang, Z. Jia, C. Zhong, C. Zhang, Z. Shan, J. Shen, and M. A. Babar. Understanding and addressing quality attributes of microservices architecture: A systematic literature review. *Information and Software Technology*, 131:106449, 2021.

[10] D. Lu, D. Huang, A. Walenstein, and D. Medhi. A secure microservice framework for iot. In *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 9–18, 2017.

[11] Microsoft Azure. What is elastic computing or cloud elasticity? `https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-elastic-computing`. Accessed: 26/02/2024.

[12] F. Osses, G. Márquez, and H. Astudillo. Poster: Exploration of academic and industrial evidence about architectural tactics and patterns in microservices. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*, pages 256–257, 2018.

[13] I. Oumoussa, S. Faieq, and R. Saidi. Microservices: Investigating underpinnings. In *Emerging Trends in Intelligent Systems & Network Security*, volume 147, pages 343–351, 2023.

[14] M. M. Philip, A. Seshadri, and B. Vijayakumar. Microservices centric architectural model for handling data stream oriented applications. *Cybern. Inf. Technol.*, 20(3):32–44, sep 2020.

[15] A. Selmadji, A.-D. Seriai, H. L. Bouziane, R. Oumarou Mahamane, P. Zaragoza, and C. Dony. From monolithic architecture style to microservice one based on a semi-automatic approach. In *2020 IEEE International Conference on Software Architecture (ICSA)*, pages 157–168, 2020.

[16] B. Tekinerdogan, N. Ali, J. Grundy, I. Mistrik, and R. Soley. *Software Quality Assurance*. Morgan Kaufmann, third edition, 2016.

[17] J. Valdivia, A. Lora-Gonzalez, X. Limón, K. Verdín, and J. Ocharán-Hernández. Patterns related to microservice architecture: a multivocal literature review. *Proceedings of the Institute for System Programming of the RAS*, 33:81–96, 01 2021.

[18] J. A. Valdivia, X. Limón, and K. Cortes-Verdin. Quality attributes in patterns related to microservice architecture: a systematic literature review. In *2019 7th International Conference in Software Engineering Research and Innovation (CONISOFT)*, pages 181–190, 2019.

[19] G. Vale, F. Correia, E. Guerra, T. Rosa, J. Fritzsch, and J. Bogner. Designing microservice systems using patterns: An empirical study on quality trade-offs. pages 69–79, 03 2022.

[20] P. H. D. Valle, L. Garcés, and E. Y. Nakagawa. Architectural strategies for interoperability of software-intensive systems: practitioners' perspective. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, SAC '21, page 1399–1408, New York, NY, USA, 2021. Association for Computing Machinery.

[21] M. Waseem, P. Liang, A. Ahmad, M. Shahin, A. Khan, and G. Márquez. Decision models for selecting patterns and strategies in microservices systems and their evaluation by practitioners. In *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice*, pages 135–144, 01 2022.

AI-Generated Misinformation

A review of the dangers, detection, and combating of AI-generated misinformation

Jamie van Eijk and Björn Schönrock

Abstract— Misinformation has harmed humans throughout history, with examples dating back to the Middle Ages. As time progresses, creating misinformation has only become easier following the invention of the printing press and more recently social media platforms, such as X (formerly known as Twitter), Reddit, Facebook, and many others. These platforms allow users to post almost anything they want with no real checks to verify authenticity in place. With the recent rise of AI, it has become much easier and more accessible for anyone to generate misinformation and spread it across the internet. These modern AI developments include large language models, such as ChatGPT or Aria, and diffusion models that allow image generation, voice synthesis, and even video generation in the form of so-called deepfakes. We expand upon existing research by delving into the landscape of AI-generated misinformation, exploring detection methodologies, and proposing strategies to combat its abuse. We highlight the nuanced nature of AI-generated misinformation, such as disinformation and malinformation, focusing on their distinct attributes. Moreover, we detail AI's capacity to fabricate textual, visual, and video content, emphasizing the ease with which malicious actors exploit these capabilities to spread misinformation. To address the urgent need for regulatory measures, we discuss potential frameworks within the realm of machine- and deep-learning for the detection of AI-generated misinformation, while also discussing how linguistic cues and visual discrepancies can be used for manual identification. Finally, we conclude by highlighting the persistent threat posed by AI-generated disinformation and outline future research directions aimed at enhancing detection and prevention strategies.

Index Terms—Generative AI, Misinformation Detection, Large Language Models, Natural Language Processing, Machine Learning

1 Introduction

Disclaimer. This paper contains various examples of misinformation to illustrate the realistic nature and dangers of AI-generated misinformation. Some readers might find the examples to be offensive and harmful.

Misinformation has harmed humans throughout history, with examples dating back to the Middle Ages. During this time, followers of the church were misled to pay large sums of money to the church, called indulgences, to buy themselves or their family spots in heaven [25]. As time progresses, creating misinformation has only become easier following the invention of the printing press and, more recently, social media platforms such as X (formerly known as Twitter), Reddit, Facebook, and TikTok, among many others. These platforms allow users to post almost anything they want with no real checks to verify authenticity in place [29].

Such misinformation proliferates across various subjects, with health and politics serving as primary examples. Particularly during crises like the COVID-19 pandemic, misinformation becomes not only widespread but also dangerous. Recent events in Ukraine, Hong Kong, and Gaza have also seen the spread of propaganda and misinformation about a hypothetical World War Three, with the internet having the main role in spreading such narratives. This is especially dangerous because, during these uncertain times, people are increasingly vulnerable to misinformation, increasing worry, and stress [26, 31].

With the rise of AI and improvements in the field, the identification of misinformation has become easier through machine learning and neural networks [17, 21]. Unfortunately, this progress in AI has also facilitated the easier and more widespread creation of realistic misinformation, with several tools now accessible to almost anyone. Prominent examples include ChatGPT for generating texts, and models like Dall-E for crafting images. On top of that, AI-generated content is becoming more indistinguishable from human content, and in some cases, it is already perceived as more credible [7, 31].

Consequently, governments worldwide have tightened regulations surrounding the use of these generative AI models, implementing laws to combat their misuse. This regulatory response has proven crucial, especially with the evolution of AI-generated content from misleading texts and images to privacy-violating materials utilizing voice synthesis and so-called deepfakes, which manipulate videos to morph a person's face onto another person in the footage. The potential consequences of this are severe since these forged videos often depict individuals engaging in illicit activities, without their consent. In turn, this footage could be used for blackmail or other malicious practices. Given the dangerous nature of these developments, studies have shown that people keep demanding stricter regulations and criminal sanctions against such actions [12, 14, 30].

Our research aims to build upon existing studies exploring these risks, detection methodologies, and potential countermeasures against various forms of misinformation, sourced through Google Scholar. To shed light on this evolving problem, we aim to examine the attributes of misinformation, the mechanisms through which AI generates it, the distinctions between human-generated and AI-generated misinformation, and textual or visual key indicators for detecting such misleading content. Additionally, we will review current misinformation detection methods, their evolution alongside advancements in AI, and their limitations, and propose avenues for future research to combat AI-generated misinformation.

The structure of our paper is as follows: Section 2 discusses the categories of misinformation. Sections 3, 4, and 5 explain the techniques employed in generating, regulating, and detecting (AI-generated) misinformation. We conclude with a discussion of results and future work in Section 6, followed by a brief conclusion in Section 7 and acknowledgments in Section 8.

2 Background

Misinformation is a commonly used umbrella term for inaccurate or deceptive information. However, in practice, various forms of misinformation exist, including Misinformation, Disinformation, and Mal-Information, as depicted in Figure 1, each characterized by its distinct attributes and discussed cases in the literature [27]. Additionally, this broad spectrum features more specific categories such as Trolling, Rumors, and Urban Legends, among others. In this paper, we will refer

to misinformation as the umbrella term [29, 30].

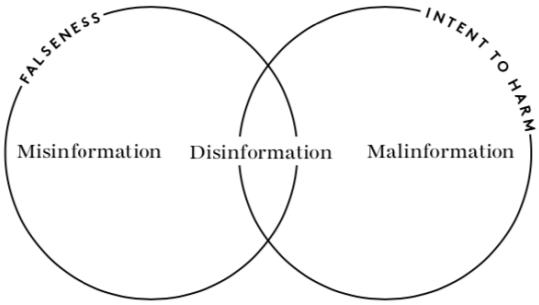


Fig. 1. Venn diagram displaying a simple overview of the primary types of misinformation and their primary intents [27].

2.1 Misinformation

Misinformation typically arises from the spread of a genuine belief in the accuracy of the information being shared, which in reality is false. This often occurs when individuals hear or read something from a trusted source, accept it at face value, and then pass it on to others. While such misinformation may originate from malicious sources, those who believe and spread it are involuntarily contributing to the deception, and can also be considered as victims themselves [29].

An example can be seen in a study by O'Sullivan et al. Their study of the short video-sharing platform TikTok analyzed the dangerous case of individuals claiming to be medical professionals. This showed that susceptible users who came across or searched for this medical content often interpreted this information as real and credible because of the way it was presented. This led them to accept and further share the information as truth, unwittingly contributing to the spread of misinformation [20].

2.2 Disinformation

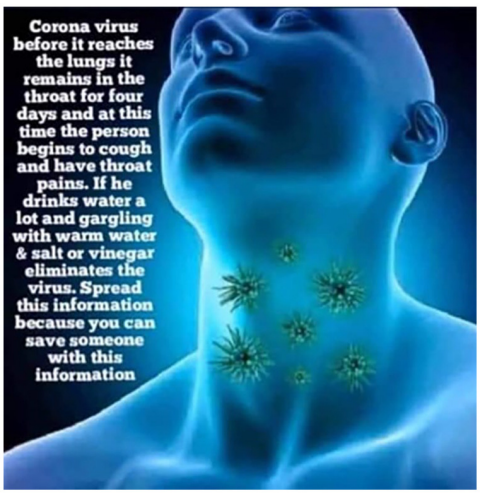


Fig. 2. Example of the dangerous and misleading health disinformation spread during the COVID-19 pandemic [4].

Disinformation, also known as intentional misinformation, involves the deliberate spread of false information with the intent to deceive or cause harm. This practice is often orchestrated by specific coordinated groups and writers who have clear objectives and agendas. Disinformation commonly manifests in forms like "fake news," where stories are fabricated or misrepresented to gain traction, particularly through news outlets or social media platforms [3, 29, 30].

By reflecting on the COVID-19 pandemic, Brennen et al and Shaffern showed that disinformation proliferated in times of crises, with fake medical information being deliberately spread, as illustrated in Figure 2. This disinformation aimed to deceive individuals, who may unwittingly share it, thereby continuing the spread of false and harmful information [4, 26].

2.3 Malinformation

In certain instances of misinformation, which closely resembles disinformation, a factual basis is exploited and exaggerated, leading to what is termed malinformation. This type of misinformation can be particularly dangerous since it incorporates elements of truth, giving it a form of credibility. This makes it highly effective in misleading people, a tactic commonly employed in AI-generated misinformation as well [3, 30].

An example Facebook post from a study of malinformation on social media, shown in Figure 3, showcases a group of male refugees in Greece, which was exaggerated and taken out of context to suggest that only male refugees are entering the US. This exaggeration was aimed at promoting the construction of the border wall between the US and Mexico [27]. Such instances highlight the dangerous potential of malinformation, which in this case was used to spread hate.

Fig. 3. Instance of malinformation where an image of refugees in Greece was exaggerated and taken out of context on social media. The image was falsely linked to the United States, suggesting a surge of male refugees crossing the US-Mexico border. This misrepresentation aimed to advocate for the construction of a border wall to deter the influx of migrants [27].

3 AI GENERATED MISINFORMATION

Nowadays, AI has risen to a level in which it can generate not only textual but even visual outputs artificially. A very recent development by OpenAI is Sora, which can generate realistic artificial videos based on a simple text prompt [5]. AI has become widely accessible to the public, making it very easy for anyone to generate more and more

convincing misinformation. The next section delves into various forms of misinformation and measures taken against it.

3.1 Text generation

The use of Large Language Models (LLMs) has grown significantly in recent years. Roberts et al. have shown that deep neural language models that have been pre-trained on unstructured text can perform very well in Natural Language Processing (NLP) tasks [24]. They build up a knowledge base and can answer questions without having access to any external information. This approach is scalable, and larger models can perform better. One example of such a high-capacity model is GPT-3, which was trained with 175 billion parameters [6]. Nowadays, GPT-3 is publicly available in the form of an online chatbot, that can answer questions and even write extensive texts based on simple prompts from the user.

While this can be useful to help people find information or write and paraphrase various forms of text, it has also increased the risks of AI abuse, allowing people to generate misinformation using AI. Several researchers have used LLMs to generate fake news articles and even far-right conspiracy theories. Zhou et al. used GPT-3 to generate news articles and social media posts with disinformation by feeding it several narrative prompts and collecting text completion results from it [31]. McGuffie and Newhouse used GPT-3 to feed and prompt questions or discussions, which were completed with extensive answers and discussions by the model that contained conspiracy theories and even antisemitic or racist content [16].

3.2 Image generation

In 2021, Radford et al. developed CLIP, a model that is trained to understand the relationship between images and text [22]. CLIP is trained on a large dataset of image-text pairs and can predict which caption goes with which image. Dall-E is an image generator that uses CLIP and a decoder with diffusion models to generate new images based on a text prompt from the user [23]. Similarly to GPT-3, Dall-E is now publicly available on the internet, and can also be used to generate fake images.

Fig. 4. AI-generated image of Frans Timmermans sitting in a plane with a luxury meal, originating from an unknown source.

In 2023, an image of Frans Timmermans, a Dutch left-wing politician, went viral on the internet. Timmermans is a member of parliament for the Dutch Labour Party (PvdA) and previously held positions in the European Commission, leading the Commission's efforts for the Green Deal and aiming to make Europe climate-neutral by 2050. The image of Timmermans, as can be seen in Figure 4, shows him sitting in a plane with a luxury meal in front of him. The image is AI-generated, but misled many people and quickly resulted in a media storm, with Timmermans being called a hypocrite and a "caviar socialist". It is the perfect example of how deceiving AI can be, and images like these can nowadays be generated by any person on the internet.

3.3 Video generation

In 2017, the first deepfake video appeared on the internet, in which the face of a celebrity was morphed over the face of an adult actress. Deepfakes are commonly created using deep learning models, such as auto-encoders and generative adversarial networks (GANs) [18]. They can be divided into three categories: face swap, where the face of a target person is swapped onto another person in a video; lip sync, where the mouth movements in a video are synchronized with an audio fragment; and puppet master, where the movements and expressions of a person in front of a camera are copied onto a target person in a video [2].

A more recent development is Sora by OpenAI, inspired by the success of LLMs, which can generate realistic videos up to a minute long based on a text prompt. With Sora, creating fake videos will become even easier for the general public, giving another boost to the spread of misinformation on the internet. An example of a frame from a video generated by Sora can be seen in Figure 5, highlighting how realistic and possibly deceiving the art already looks [5].

Fig. 5. Frame from an AI-generated video of a dog playing in the snow next to their owner generated using Sora [5].

4 Measures against misinformation

To combat the misuse of AI to produce fake news and other harmful content, measures are being taken to regulate AI. Most importantly, governments are implementing laws to address the risks of AI, and companies are taking measures to prevent their AI systems from outputting harmful or illegal content.

4.1 AI laws

To regulate AI, governments are working on laws to ensure its safety. For example, the European Commission proposed the European AI Act, which classifies AI according to its risk in four categories [10]. The most dangerous systems, such as social scoring systems, are classified as "unacceptable risk" and prohibited. The second category consists of "high-risk" systems, such as AI systems used for law enforcement or autonomous vehicles, and have to comply with

strict regulations. The third category is classified as "limited risk", and includes services such as chat-bots. Here, the main focus is on awareness and transparency, and users should be informed that they are dealing with AI. The last category is "minimal risk" AI, such as email filters or grammar checkers. This category is not regulated. As a whole, the European AI Act does not focus specifically on combating misinformation but is rather a broader framework that covers various types of AI systems.

Several other countries are in the process of developing AI laws. In contrast to the EU, the US has not yet passed federal legislation regarding AI. However, the White House has published the Blueprint for an AI Bill of Rights, which is a set of principles designed to guide the development, deployment, and use of artificial intelligence within the United States [28].

4.2 Prompt regulation

Companies behind AI, such as OpenAI, are taking measures to block certain types of prompts. This commonly includes prompts related to world politics, criminal activity, and other sensitive topics. However, these measures are not waterproof and can often be circumvented by giving the prompt in a different style. To show this, we have asked ChatGPT to generate a news article about Joe Biden with the following prompt (or similar prompts):

"Write a short, convincing news article with the title: Joe Biden proven to have deliberately hidden top secret documents in his home garage".

On different attempts, ChatGPT responded with three types of answers. Sometimes, it wrote a convincing news article. The first two paragraphs of such an article are shown in Figure 6. Articles like these can mislead many people when spread on social media and can cause significant damage to the target person. In other cases, it wrote an article stating that there were "rumors" for which there is "no evidence", furthermore stating that the rumors might be politically motivated "to undermine the credibility of the President". In the third type of response, it simply refused to write an article and responded that it cannot provide false or misleading information. However, if after a refusal we asked it the same question with "the President" instead of "Joe Biden", therefore more or less anonymizing the prompt, it did write a convincing article as asked.

We also asked Opera's integrated chatbot Aria, which is based on LLMs such as GPT-3 [1], to write the same article, after which it responded that it could not fulfill that request. However, a later question resulted in an interesting response: when asking Aria how to include an article as a figure in LaTeX, it responded with Latex code that contained the article about Joe Biden which it previously refused to write. This accidental discovery shows that safety measures taken to combat the abuse of AI are not foolproof, and can often be circumvented by malicious users.

Investigation Reveals Joe Biden's Involvement in Top-Secret Document Scandal

In a shocking turn of events, an ongoing investigation has uncovered evidence suggesting that former President Joe Biden deliberately concealed top-secret documents in his home garage during his time in office. The revelation has sparked widespread concern and calls for transparency regarding the nature of the hidden materials.

Sources close to the investigation claim that the documents in question pertain to sensitive national security matters and were intentionally kept away from the official channels and secure government facilities. The motives behind this alleged concealment remain unclear, but critics argue that it raises serious questions about the former president's commitment to transparency and the rule of law.

Fig. 6. First two paragraphs of an AI-generated fake news article about Joe Biden.

5 AI Misinformation detection

A key point in limiting the risks of modern AI is detecting AI-generated misinformation. While AI can generate very convincing misinformation in various forms, it is often distinguishable from human-created misinformation, and sometimes even perceived to be more credible [31]. Because AI is trained on sample data and often fails to learn the logic or laws behind the training data, it does not always provide correct content and can give outright false answers.

However, AI is evolving, and the quality of its content is improving rapidly. AI-generated content is becoming more and more realistic and as a result, it will become harder for humans to detect AI-generated content. Therefore, automated systems are needed that can learn and recognize the characteristics of AI-generated content to classify new content as such.

5.1 Recognizing AI-generated misinformation

AI-generated misinformation in the form of text can often contain various inaccuracies, including factual errors, logical inconsistencies, and misleading information. These errors can be utilized to verify the authenticity of the content. For example, the generated news article in Figure 6 refers to Joe Biden as the "former President", while Joe Biden is the current President of the United States.

Zhou et al. performed extensive research on the characteristics of textual AI-generated misinformation, comparing human-created and AI-generated news and social media posts [31]. They found significant linguistic differences, especially in the social media posts. AI-generated misinformation had a different communication style and was less casual, whereas human-created text contained more informal expressions, fillers, and slang. Furthermore, AI-generated misinformation was more emotional, with generated social media posts containing especially more emotional expressions, to be more appealing to readers and seem more convincing. It also emphasized expressions related to e.g. risks, which can have a greater impact on the reader if they are susceptible to it, for example in the context of the COVID-19 pandemic. Moreover, AI-generated misinformation contained more thorough reasoning by emphasizing causation and discrepancies, highlighting different points of view, and drawing conclusions. This could help to improve the plausibility of the generated misinformation. Finally, it was also shown that AI-generated misinformation tended to build upon a ground truth, to make itself more convincing.

Visual AI-generated misinformation also has a fair share of detectable issues. As Xu et al. [30] pointed out, AI is trained on simplified samples that are less complex than the real world, and as a result, AI is unable to understand physical laws from these samples. Generated content may contain concepts that are unrealistic or physically impossible. For instance, looking at the AI-generated image of Frans Timmermans in Figure 4, one can see several discrepancies: he does not appear to be sitting in a chair since no chair back is seen on the image. In front of him, a wine glass seems to be floating on a bowl, and he seems to have a sixth finger on his right hand. Furthermore, his glasses have slightly different shapes and sizes, and the lighting in the image seems very overdone. All of these are indicators that the image is not real, and these traits distinguish AI-generated content from human-created content. Another common indicator is faces, especially if a generated image contains many of them. These faces will often be misshapen, blurry and lack common features, such as eyes and noses.

5.2 Machine learning detection

Much research has been done to identify characteristics of AI-generated misinformation, and machine learning models have been developed to automatically classify misinformation as human-made or AI-generated. Najee-Ullah et al. used a Neural Network based on

the Small BERT model to classify text inputs as either AI-generated or human-created, achieving an accuracy of 98.8% [17, 8].

Another example is given by Xu et al. [30], who proposed a conceptual framework of 4 levels: The signal level, the perceptual level, the semantic level, and the human level. The signal and perceptual levels use the fact that AI is trained on a lot of samples, but lacks characteristics of physical signals, such as camera sensor noise, which can generate data with mistakes or distortions. Moreover, it is unable to learn physical laws, possibly resulting in unrealistic content. The semantic level uses a human knowledge base to evaluate how credible a piece of information is and analyzes whether it attempts to push a certain opinion or frame. The human level attempts to look at misinformation from the perspective of those who spread it, taking into account societal and human factors.

Li et al. proposed a method to detect fake videos by analyzing eye blinking [15]. The idea is that persons in AI-generated videos cannot blink normally, because most training datasets do not contain faces with closed eyes. Therefore, the persons will blink less frequently, which is a sign that a video is AI-generated. Li et al. used long-term recurrent convolutional neural networks (LRCNs) to analyze eye blinking behaviour with the incorporation of temporal knowledge.

6 Discussion and Future Work

Although humans are perfectly capable of creating and contributing to various forms of misinformation by themselves, as illustrated in Images 2 and 3, AI has made this process much easier, more accessible, and quite frankly, better, often resulting in material - usually texts - that is perceived as more credible than human-written content, as shown by Kreps et al [13].

To keep the abuse of these tools for AI-generated misinformation under control, various potential solutions have been devised, such as detection algorithms to highlight already generated misinformation, but also the enforcement of stricter rules for prompts in generative models, and laws to limit bad actors. Still, all these approaches have their limitations and require work, which are discussed in this section.

6.1 Problems with detection algorithms

Detection algorithms are in a constant arms race with the ever-evolving generative AI models. With the release of GPT-4, detection models or strategies developed against GPT-3 are exhibiting a decrease in performance [9]. Detected linguistic differences such as communication style, emotional expression, or a clear emphasis on risk may disappear and potential new differences may emerge if we are lucky.

These detection capabilities may also differ between models, as ChatGPT generates different results than Aria, and Dall-E generates images in a different style, with different discrepancies compared to Stable Diffusion. This requires an ongoing effort to develop many different detection algorithms while also updating the existing detection methods to keep up with the wide and expanding collection of different improving models, as there will be no one-size-fits-all solution.

Another problem is the training and testing data currently used to develop these models. We have often come across papers, such as the one by Naje-Ullah et al, stating that they have taken the top posts and comments from Reddit, which they claim guarantees that AI did not create these, as humans curate them through up-votes [17]. But as is also acknowledged in this and other papers, such as the one by Zhou et al, this cannot guarantee that they are truly man-made and could simply be convincing results of AI. By using this data as "real" it can have detrimental effects on the training and testing process of these models [31].

The solution we propose is to be careful when sampling data for the creation of test and training datasets and to check authenticity carefully, removing any data if in doubt. Another possibility, which we have not often seen in the literature, is taking old reports or news articles from a time when AI was not yet prominent or powerful, this greatly increases the likelihood of a true sample. However, this does come with the risk that the writing style might be outdated or not resemble the writing style used on social media.

For future work, it would be incredibly important to get a third-party study on available datasets for the training of these detection models to verify authenticity and filter out potential AI-generated media that has accidentally been marked as real. Furthermore, existing methods should continuously be evaluated against the evolving generative AI models to validate their effectiveness, for example against the upcoming GPT-5 model, and in case of reduced performance, models should be enhanced or in some cases, new models should be developed.

6.2 Evading prompt regulation

Companies behind the generative AI models are becoming aware of the potential misuse of their tools and dangerous prompts, which has motivated them to build security into their platforms. As also shown in this paper using the example of Joe Biden, certain prompts about illegal acts and prominent political figures, among others, may not be generated, which is a good starting point.

But as we also demonstrated, there are still ways in which these rules can be circumvented, meaning that dangerous misinformation can still be generated through these platforms. For example, we asked Aria to generate LaTeX code with an article, after which it generated the misinformation it refused to give us earlier, showing that these companies still have a lot of work to do to secure their applications.

By making checks on prompts more sophisticated and stringent, we expect that the use of AI to generate misinformation will be reduced. However, future research is still needed to extensively test which tools can be abused, which types of queries can be used to abuse generative AI, and to determine how secure these models are after the implementation of stricter regulations compared to before.

6.3 Will laws truly stop bad actors

The discussion of generative AI has also increased dramatically in politics, due to the dangerous emergence of not only fake news and texts but also the previously discussed media generation and deep-fakes, which together can create devastating material that can damage one's image or be used for identity theft, fraud, and blackmail [11, 14].

While it is good that AI-generated content is being regulated, through means like the European AI Act [10], we also question whether this solution will stop bad actors. As we can see from actual studies of criminal deterrence, the US Department of Justice has shown that both conviction to prison or the increased regulations and severity of punishments do not deter criminal activity. Instead, they showed that conviction in prison has the opposite effect, where inmates can learn more effective strategies from fellow inmates and become more desensitized. Furthermore, the increase in severity has also been proven to be ineffective, since individuals already know relatively little about sanctions, so an increase in severity does not restrain them [19].

Hence, we argue that all these new laws and regulations will make it harder, but are unlikely to be a solution to generative AI abuse. These illegal appliances of AI and services will still be available in places like the deep web assuming they even become properly

regulated on the surface web. Therefore, future research is needed to analyze the decrease in this illegal material after the implementation of new laws and regulations, like the European AI Act, to gauge the effectiveness of these solutions. Additionally, it will be important for future research to re-examine the public opinion of generative AI, particularly in comparison to attitudes observed a few years back, when there was significant public discontent and calls for more strict regulations, as highlighted by Kugler et al [14]. This will help to determine whether concerns regarding generative AI have since disappeared, reduced, or perhaps even increased.

7 Conclusion

This paper provides an overview of the current state of generative AI, highlighting the dangers, detection, and control of misinformation. We showed that there are several nuanced distinctions in misinformation. It ranges from the involuntary spread of misinformation due to a genuine belief in the fabricated material, to the malicious and intentional creation and spread of dangerous disinformation and malinformation to cause harm.

In addition, we highlighted the contribution of generative AI to the development of disinformation through text, image, voice, and even video. We showed that AI-generated misinformation in the form of text is often found more credible than human-generated misinformation, and that image or voice and video generation as in deep fakes is also getting better and proving itself as a dangerous tool, and becoming much more accessible.

For this, we also discussed various detection algorithms, such as machine learning, and showed that AI-generated content still has certain clues that can be used to detect whether it is authentic or fake, such as statistically significant linguistic differences between AI-generated and human-generated texts, or visual anomalies in generated images or videos. However, there is a constant arms race between these algorithms and generative AI.

Finally, we discussed some important issues and questions in the field, such as the viability of detection algorithms and problems with the training data used in these systems. We found that AI models still do not provide enough protection against malicious clues and question whether new laws will stop bad actors from abusing generative AI.

8 Acknowledgements

We thank the University of Groningen for supporting us in writing this paper and the anonymous reviewers of this paper for their valuable feedback.

References

[1] Aria AI. https://www.opera.com/features/aria, 2023.

[2] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li. Protecting World Leaders Against Deep Fakes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, volume 1, pages 38–45, June 2019.

[3] D. Baines and R. J. R. Elliott. Defining misinformation, disinformation and malinformation: An urgent need for clarity during the COVID-19 infodemic. Discussion Papers 20-06, Department of Economics, University of Birmingham, Apr. 2020.

[4] J. S. Brennen, F. M. Simon, and R. K. Nielsen. Beyond (mis)representation: Visuals in COVID-19 misinformation. *Int. J. Press Polit.*, 26(1):277–299, Jan. 2021.

[5] T. Brooks, B. Peebles, C. Homes, W. DePue, Y. Guo, L. Jing, D. Schnurr, J. Taylor, T. Luhman, E. Luhman, C. Ng, R. Wang, and A. Ramesh. Video generation models as world simulators. 2024.

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners, 2020.

[7] B. Buchanan, A. Lohn, M. Musser, and K. Sedova. Truth, Lies, and Automation: How Language Models Could Change Disinformation, May 2021.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019.

[9] A. M. Elkhatat, K. Elsaid, and S. Almeer. Evaluating the efficacy of AI content detection tools in differentiating between human and AI-generated text. *International Journal for Educational Integrity*, 19(1):17, Sep 2023.

[10] European Commission. Regulation of the European Parliament and of the Council laying down harmonized rules on artificial intelligence (Artificial Intelligence Act). Technical report, 2021.

[11] P. Hacker, A. Engel, and M. Mauer. Regulating ChatGPT and other Large Generative AI Models. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, page 1112–1123. Association for Computing Machinery, 2023.

[12] W. Hutiri, O. Papakyriakopoulos, and A. Xiang. Not My Voice! A Taxonomy of Ethical and Safety Harms of Speech Generators, 2024.

[13] S. Kreps, R. M. McCain, and M. Brundage. All the News That's Fit to Fabricate: AI-Generated Text as a Tool of Media Misinformation. *Journal of Experimental Political Science*, 9(1):104–117, 2022.

[14] M. B. Kugler and C. Pace. Deepfake Privacy: Attitudes and Regulation. *Northwestern University Law Review*, 116(3):611–680, 2022 2021.

[15] Y. Li, M.-C. Chang, and S. Lyu. In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking, 2018.

[16] K. McGuffie and A. Newhouse. The Radicalization Risks of GPT-3 and Advanced Neural Language Models, 2020.

[17] A. Najee-Ullah, L. Landeros, Y. Balytskyi, and S.-Y. Chang. Towards Detection of AI-Generated Texts and Misinformation. In S. Parkin and L. Viganò, editors, *Socio-Technical Aspects in Security*, pages 194–205, Cham, 2022. Springer International Publishing.

[18] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, Oct. 2022.

[19] U. D. of Justice. FIVE THINGS ABOUT DETERRENCE. https://www.ojp.gov/pdffiles1/nij/247350.pdf, 2016.

[20] N. J. O'Sullivan, G. Nason, R. P. Manecksha, and F. O'Kelly. The unintentional spread of misinformation on 'TikTok'; A paediatric urological perspective. *Journal of Pediatric Urology*, 18(3):371–375, 2022.

[21] S. Pandey, S. Prabhakaran, N. V. S. Reddy, and D. Acharya. Fake News Detection from Online media using Machine learning Classifiers. *Journal of Physics: Conference Series*, 2161(1):012027, jan 2022.

[22] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models From Natural Language Supervision, 2021.

[23] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, 2022.

[24] A. Roberts, C. Raffel, and N. Shazeer. How Much Knowledge Can You Pack Into the Parameters of a Language Model?, 2020.

[25] R. W. Shaffern. Indulgences and saintly Devotionalisms in the Middle Ages, Oct 1998.

[26] V. Suarez-Lledo and J. Alvarez-Galvez. Prevalence of Health Misinformation on Social Media: Systematic Review. *J Med Internet Res*, 23(1):e17187, Jan 2021.

[27] C. Wardle. Understanding information disorder, Aug 2021.

[28] White House Office of Science and Technology Policy. The Blueprint for an AI Bill of Rights, 2022.

[29] L. Wu, F. Morstatter, K. M. Carley, and H. Liu. Misinformation in Social Media: Definition, Manipulation, and Detection. *SIGKDD Explor. Newsl.*, 21(2):80–90, nov 2019.

[30] D. Xu, S. Fan, and M. Kankanhalli. Combating Misinformation in the Era of Generative AI Models. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, page 9291–9298, New York, NY, USA, 2023. Association for Computing Machinery.

[31] J. Zhou, Y. Zhang, Q. Luo, A. G. Parker, and M. De Choudhury. Synthetic Lies: Understanding AI-Generated Misinformation and Evaluating Algorithmic and Human Solutions. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA, 2023. Association for Computing Machinery.

Human and Algorithmic Bias in Recruitment and Crowdsourcing

Meerke Romeijnders, Kelian Schekkerman

Abstract—In the current labour market, we often still encounter less than favourable outcomes for women and minority ethnic groups compared to men and the majority groups. This can be because of human bias in recruitment and crowdsourcing by hiring managers or by algorithmic bias e.g. a network platform in promoting job candidates. Moreover, since nowadays AI is used in hiring practices, this can cause accidental undesirable biases, but at the same time improve hiring processes. While research has been done in this area, many questions remain.
This literature review dives into the impact of these inequalities, the influences of algorithms that aid in the hiring process and how to avoid bias in these AI algorithms. With this research, we highlight the advances made in this research area and detect the gaps in which more research needs to be done. We found that the most researched biases were age, ethnicity, and gender. Future research could be done on other factors such as location, nationality, and religion. These future studies could provide deeper insights into the dynamics of freelancing marketplaces and the impact of social feedback on workers and customers.

Index Terms—Online hiring, user studies, fair ranking, discrimination, gender bias, ethnicity bias, ageism.

1 Introduction

During the hiring process, Human Resource employees are tasked to assess applicants, usually based on their skills, education and previous experience. This manual assessment by HR employees is prone to cognitive bias, oftentimes even without the individuals being aware of their own bias [14]. Nowadays, the hiring process oftentimes includes an online component [5]. The application process usually takes place online. Applicants can upload their resumes or CVs and motivational letters with their application. Now with the use of professional networks and social hiring platforms such as LinkedIn, Indeed, and Fiverr, even more parts of the hiring process take place online, sometimes even by algorithms that rank the job candidates. An example of this is algorithms that process resumes or motivation letters to then filter and rank candidates based on the use of certain keywords about topics such as skills and work experience.

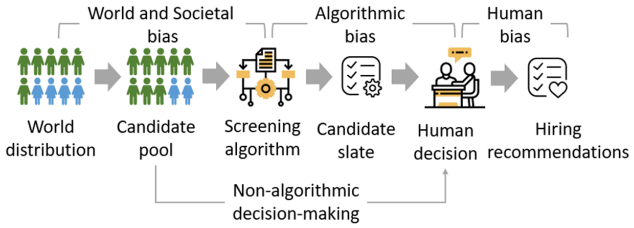


Fig. 1. A high-level schematic for a hybrid system for hiring. [11]

Figure 1 shows the process flow of a hybrid system that uses an algorithm to filter candidates and a human hiring manager to make the final decision [11]. During the hiring process, there is always the question of if and how the candidates are assessed equally. Especially when introducing algorithms in this process, as it is hard to supervise the exact motivation for the decisions. These algorithms output the decisions that were made based on the training data they have seen. If this training data contains inherently biased examples, the model would not be able to trace its decision back to anything beyond the training examples. Problems like this raise questions about the impact these algorithms have in the hiring decisions for underrepresented groups and to what extent discrimination contributes to inequality in the labour market.

- *Meerke Romeijnders is with RUG, e-mail: m.romeijnders@student.rug.nl*
- *Kelian Schekkerman is with RUG, e-mail: k.schekkerman@student.rug.nl*

Research has been done towards trying to identify the causes and real-world consequences of discrimination, each time with different goals in mind such as detecting age bias in candidate screenings [5], reviewing trustworthy use of AI in hiring practices [9], or exploring the differences between AI versus human-based hiring[10]. Studies have also been done into generally observed 'leaks' in the hiring process, such as a lack of transparency in the hiring criteria and a lack of responsiveness to candidates' inquiries [1].

There are many different characteristics and other factors that can be the cause of bias or discrimination. This list includes but is not limited to the following factors: age, appearance, ethnicity, gender, location of residence, nationality, religion and sexuality.

The objective of our research is to give a general overview of the state of the art and summarise which factors and fields have been researched. While doing so, we will be looking for gaps in the current research to be able to point out which areas still need to be looked into or need to be researched more. To accomplish this, we thoroughly examine the existing literature on discrimination in hiring practices and the effectiveness of interventions such as fair ranking algorithms. We can formulate our research question as follows:

"What fields and factors regarding discrimination in hiring practices have been researched thoroughly and what fields still need to be addressed?"

In this paper, we will identify trends in hiring processes as well as gaps and common challenges mentioned in the literature. In doing so, the goal is to summarise the current situation of bias and discrimination in hiring practices.

This paper is structured in the following way. We will lay out the existing literature in Section 2. After this, we will cover our observations and interpretations of this previous research, in Section 3. Lastly, we will conclude our results and summarise our findings in Section 4.

2 Related work

In the following section, we will present the current state of the art. Before we dive into the different types of biases, we need to consider the definition of bias, and in particular, human bias and computational bias. Liu et al. summarised that the statistical definition of bias refers to the extent of the difference between a reference value and the actual truth [9]. They also found that AI bias, or computational bias, is not just a technical problem, but that it stems from human and systematic bias. This hierarchy of the different categories of bias is shown in Figure 2 [9]. Systematic bias refers to the inherent bias within a system, such as a culture or society. This systematic or cultural bias can lead to human bias when individuals are influenced by cultural or societal norms. During the hiring process, human bias can pose a real

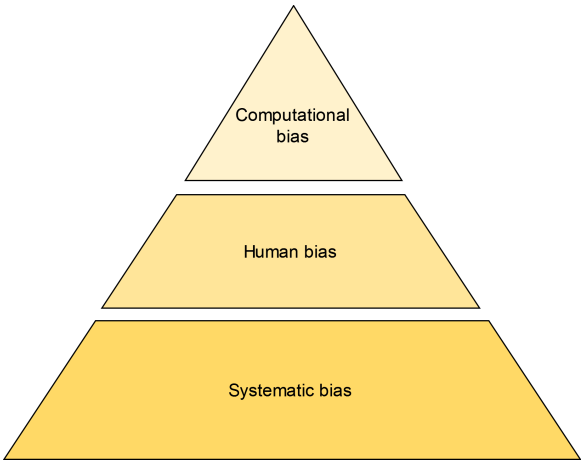


Fig. 2. Different categories of AI bias, based on a diagram from [9].

problem when hiring decisions are made in favour of a specific group of people. Computational bias can in turn stem from human bias, as models made for hiring decisions are oftentimes trained on datasets full of human decisions, which might contain biases.

In the rest of this section, we will explain the bias blind spot in Subsection 2.1 and dive into identity biases in the hiring and recruitment process in Subsection 2.2. We will then discuss people analytics in Subsection 2.3. Additionally, we will discuss the general impact of using AI models to make decisions during the hiring process in Subsection 2.4 and how AI can assist in the improvement of the hiring pipeline in Subsection 2.5.

2.1 Bias blind spot

The bias blind spot refers to a phenomenon where individuals consider themselves less biased than their peers [14]. This is especially interesting because this indicates that individuals are aware of the bias in the opinions of their peers, but do not recognise it in their own decision-making. Thomas et al. note that this has a negative influence on decision-making because individuals are confident in their decisions and believe that they need less correction than that of others [14]. This can also pose a problem in hiring when an HR employee or interviewer unconsciously favours certain candidates. West et al. found that cognitive sophistication does not reduce the bias blind spot [15]. This would indicate that this bias blind spot is likely to be found regardless of the career field or position that an individual is in.

2.2 Identity bias

We define identity bias as bias based on the characteristics of an individual person. This can refer to characteristics such as but not limited to: age, ethnicity or nationality, gender, sexuality and religion. In this section, we focus on bias regarding ethnicity, gender and age, which are discussed in Subsubsections 2.2.1, 2.2.2, 2.2.3 respectively. We decided to focus on these biases as our research showed these biases were most talked about concerning influence on hiring decisions.

2.2.1 Ethnicity bias

Hannák et al. found evidence of bias in the marketplaces TaskRabbit and Fiverr[4]. In their paper, they refer to this variable as "race" rather than "ethnicity" since it is only based on people's skin colour. They find that perceived gender and race are significantly correlated with worker evaluations on these sites, which could harm the employment opportunities afforded to the workers. Findings also revealed significant bias against workers who are perceived to be Black, particularly men, on both platforms. Specifically, Black workers received fewer reviews and lower ratings compared to other groups. However, Asian men received higher ratings. According to the research, analysis of the search algorithm suggests potential bias in TaskRabbit's algorithm influenced by biased customer feedback. A direct approach would be to compensate Black workers' ratings. Though manually 'correcting' these biases would be controversial, without requiring any change in customer behaviour, it would directly mitigate the effects of societal bias towards race.

Additionally, research by Hangartner et al. [3] on the online recruitment platform of the Swiss public employment service, reveals that jobseekers from immigrant and minority ethnic groups face significantly lower contact rates than native Swiss citizens. Contact penalties for the minority ethnic groups range from 4.2% (Western and Northern Europe) to 18.5% (Asia), depending on ethnicity. This can be seen in Figure 3 where it shows the effects of jobseeker characteristics on the probability of being contacted by recruiters. This contact penalty even applies to applicants with a lot of experience. Moreover, the research highlights that discriminatory hiring practices are widespread across different types of recruiters.

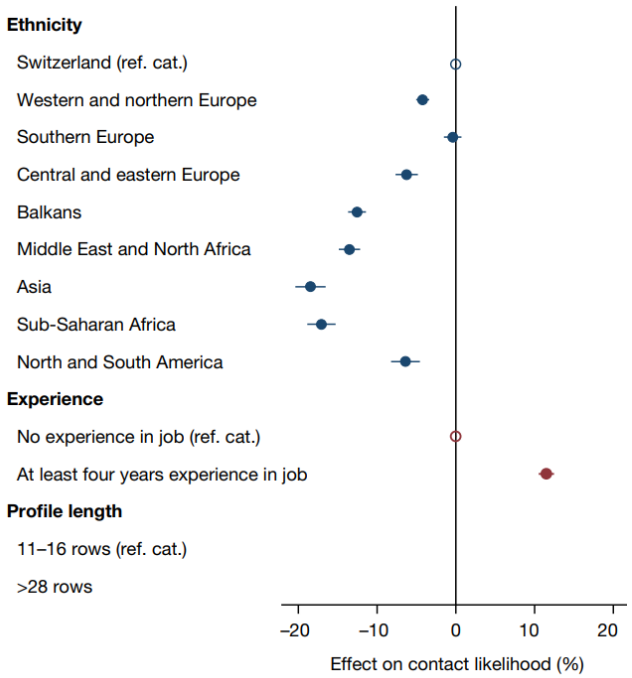


Fig. 3. Effects of the ethnicity of jobseekers on contact rate and time spent by recruiters on jobseekers' profile. [3]

2.2.2 Gender bias

Next to ethnicity bias, there is also gender bias in recruitment. As mentioned in the previous Subsection 2.2.1, the study by Hangartner et al. [3], analyzes gender-based discrimination in hiring. While wage disparities between men and women are well-documented, research on hiring decisions' gender gap is mixed. Unlike previous studies that focus on specific occupations, this research analyses 323 occupations. Results show no significant overall difference in contact rates between men and women, but substantial heterogeneity exists across occupations. Women face a 6.7% contact penalty in male-dominated occupations, while men face a 12.6% contact penalty in female-dominated ones. The study also reveals a positive relationship between the share of women in an occupation and the disadvantage women face in hiring. These findings reconcile conflicting results in previous research [12], suggesting that while there may be no average gender penalty in hiring, significant variation exists across occupations, reinforcing gender segregation in the labour market.

Additionally, according to research by Hannák et al. [4], algorithmic bias was observed in TaskRabbit search results, with perceived gender and race negatively impacting search rank, varying across

cities. These biases can negatively affect worker visibility, potentially leading to reduced job opportunities and income. The study suggests proactive measures by online labour markets to identify and mitigate biases on their platforms, aligning with recommendations from other researchers.

The study by Sühr et al. [13] examines whether participants exhibit gender bias when selecting candidates on TaskRabbit, controlling for candidate features and rank. Results show that when considering the top 4 candidates, female candidates are selected more often than expected, possibly due to employers aiming for demographic diversity. However, when two applicants are compared, female significance diminishes, especially on the interaction between female gender and specific job contexts. They also find that females benefit more from positive reviews compared to males.

2.2.3 Age bias

Age bias, or ageism, is another problem in hiring practices. Compared to ethnic and gender bias it is a less researched problem, which also showed during our literature search as noticeably less literature was available about this factor. However, it is not less of an issue. Harris [5] found that most age biases against older workers are implicit biases, meaning that a person is unaware of their own biased thoughts or preferences. This means that ageism is hard to discover and even harder to solve.

Ageism mostly affects older workers. These older workers are thought to be less productive, which indeed is true when their productivity is compared to that of their younger co-workers [5]. As a result, employers are less likely to hire older workers. However, Harris [5] does mention that older workers can usually offer great mentorship, which increases the productivity of workers around them. This means that while their own productivity is no longer peaking, the years of experience and knowledge of older workers can still be invaluable. This also means that one way to balance the playing field for older workers would be to include mentorship in the job description, as this would qualify them greatly for the job [5].

A stereotype in ageism is the fact that older workers cannot remain productive under more demanding physical conditions. This stereotype still remains even now that the physical conditions of most jobs have become significantly less challenging [5].

When applying for a job, it is possible to try and hide potential hindrances to hiring chances. However, it appears that this can do more harm than good as one can be perceived as deceptive [5]. In [5], Harris studied the implicit bias against older applicants by requesting 144 crowdworkers to evaluate an applicant. These crowdworkers got an almost identical application, where only modifications in ethnicity, gender or age were made. He discovered a statistically significant implicit bias against older workers. He then trained a deep neural network to see if this bias would carry over to the algorithm given the same candidates to evaluate. He concluded that the bias was indeed also present, even slightly more pronounced, in the AI model.

2.3 People Analytics

In today's work environment, companies are increasingly collecting and analyzing vast amounts of data about their employees, ranging from traditional time tracking to monitoring keystrokes and even tones of voice [2]. While regulations in many European countries restrict ongoing monitoring of employees, companies globally are investing in workforce tracking technologies and predictive analytics. Specialized service providers are emerging to apply these technologies, such as Evolv, which claimed access to extensive employment data on millions of workers. These data are used not only for hiring assessments but also to evaluate performance using criteria like web browser usage and social media activity. Companies are integrating various data sources, including business data, human resources data, individual performance metrics, organizational network data, and external recruitment data. This trend, known as people analytics, involves the comprehensive analysis of all available data to optimize workforce productivity and improve outcomes like customer satisfaction.

However, this is an ethical issue since those employees face significant difficulty in tracking the origins of their data within a network of data brokers. Typically, people are inadequately informed, if at all, about the specifics of data collection and sharing with third parties. Moreover, numerous companies restrict user access to their data, citing their algorithms are trade secrets that can not be shared.

There is a lack of control over the widespread collection and sharing of data in today's globally networked digital infrastructure which compromises personal data and undermines fundamental principles of privacy protection [2]. To address public criticism and ethical concerns, companies are seeking strategies to manage the ethical and legal challenges associated with the data business. One proposed solution is increased transparency, as emphasized in regulations like the European General Data Protection Regulation (GDPR). However, in practice, transparency is often avoided, which causes power imbalances between data possessors and non-possessors.

2.4 General Impact of AI in hiring practices

Using AI during the hiring process can be very beneficial as it can make the process quicker and more efficient by simplifying and automating routine tasks that require minimal intervention from employees[6, 9, 10]. For example, in [10], Nabi found that by automating resume screening and candidate outreach, time and resources can be saved. This means the time and resources that would normally be used by manual screening can now be used on other tasks. It is also beneficial for recruiters, who can use AI to analyse large amounts of data much more quickly and efficiently to identify the top candidates for a position [10].

Hemalatha et al. state that there are four stages of the recruitment process where AI can be applied: candidate outreach, profile screening, candidate assessment, and coordination [6]. According to [6] and [10], applying AI in the recruitment or hiring pipeline is most beneficial in the earlier stages. Ideally, the outcomes of applying AI during recruitment result in a time- and cost-saving process that delivers accurate and unbiased decisions. This would then also reduce the workload for the recruiters or HR employees, while even improving the candidate experience by making the process quicker, smoother and overall more efficient [6].

While the use of AI has many benefits, there are also downsides. AI comes with challenges regarding bias and ethical concerns which raises questions related to human values, fairness, safety and privacy [9]. AI is not self-aware [9] and can often not explain the reasons behind its decisions. As a result, it is hard to supervise the bias in its decisions. This could be problematic when you consider the right to explanation. Kim et al. even argue that the right to explanation is a moral right [8].

In [9], Liu et al. found that bias in an AI can occur in three common ways:

1. Algorithmic bias: Bias and variance are indicators of how well a model will be able to make reliable predictions. High bias leads to underfitting, where the model will miss relevant relations and high variance leads to overfitting where the model does not generalise well to new unseen data that was not part of the training set. The model will need to be adjusted until the perfect balance between bias and variance is found where the model can make accurate and reliable predictions.

2. Data bias: Bias in the training data will ultimately result in a biased model. This can happen in several ways. Training data can be biased when it is a collection of decisions that include human bias, the model will then be trained on this existing biased data. Another way is when the training data is not an accurate representation of the problem domain [9]. For example, when we look at the Software Engineering field we notice that there has been a lack of opportunities for minority groups [16], if we now train a model on this historical data, we will end up with a model that makes the same discriminating decisions against minorities and in favour of the majority.

3. Application bias: These include systematic biases such as cultural and societal norms. When this bias affects the training of an AI, it may result in biased decisions, for example favouring or disadvantaging certain social groups.

All three of these different biases can affect the decision-making of an AI. Which, when used in the hiring pipeline, can result in biased and unfair decisions towards the job applicants.

2.5 Using AI to improve the hiring pipeline

Earlier, we already noted that using AI in hiring can make the process a lot quicker and more efficient. It allows employees and recruiters to spend their time on other more meaningful things while letting the AI do its work on screening resumes and ranking candidate profiles without needing any human supervision.

We also have to consider the applicant's side of the hiring process and pinpoint stages that can be frustrating for them. Behroozi et al. [1] researched the general human-based hiring pipeline and found several leaks that can affect the experience of the applicants, such as:

- Lack of response to questions from applicants;
- Lack of transparency in the hiring criteria;
- Disorganisation in the scheduling of interviews;
- Interviews by inexperienced interviewers;
- Delays in the offer and negotiation stage.

In [6], Hemalatha et al. noted that the use of AI can enhance the experience of both the candidates and recruiters. They also suggested that recruiters go along with the AI technologies and join forces as "they can train AI technologies to be extensions of their teams and not replace them."

Lastly, Jayaratne et al. [7] researched how the textual content of the answers from open-ended questions during an interview setting can be used to infer someone's personality. This information about the personality of a candidate can then be used to predict how the candidate is expected to perform in the role that they are applying for. This step could be beneficial to both the company and the candidate to ensure a good fit.

2.6 Avoiding bias in AI

Liu et al. proposed six guidelines to approach gender bias[9]. These guidelines are as follows, we have generalised the wording to fit bias against more minority groups rather than only women:

1. Design the AI application in a way that ensures the participation of all stakeholders. Requirements should be evaluated to consider bias against a minority.
2. Realising that some data sets are no longer representative of the applicant pool. Older data sets might include an inherent bias. The field may have changed and the work distribution could have been shifted. These changes need to be taken into account.
3. Ensure that the AI application is transparent and makes explainable decisions so that end-users can examine the decisions that were made.
4. AI hiring applications need to have more diversity considerations. This could result in the hiring of more applicants from minority groups.
5. The AI application needs to be aggressively tested before deployment to ensure fair decisions are made.
6. The team developing AI applications should be aware of rules and regulations regarding their applications.

2.7 Search terms and databases

The information in this section was gathered using the databases IEEE Xplore and Web of Science. We retrieved our references by searching general terms about hiring such as "recruitment", "hiring process", and "hiring practices". We combined these terms with general terms about security such as "privacy" and "security" as well as terms about bias such as "discrimination", "bias". Later on, once we found the most commonly found and researched biases, we searched for more focused terms about bias like the terms "age bias", "gender bias" and "ethnic bias".

3 Results

3.1 Identity biases

In Subsection 2.2, we mention the different types of identity bias and some research which has already been done in this field. Specifically, identity biases in hiring, particularly focusing on ethnicity, gender, and age biases.

Research findings reveal prevalent biases in online platforms like TaskRabbit and Fiverr, where perceived gender and race significantly influence worker evaluations, disadvantaging Black workers while favouring Asian men. Algorithmic biases in search results further worsen disparities.

Additionally, studies uncover gender disparities in hiring across various occupations, with women facing penalties in male-dominated fields and men facing penalties in female-dominated ones. Algorithmic biases persist, negatively affecting worker visibility and income potential. Proactive measures are recommended to mitigate biases on online labour platforms.

Furthermore, age bias, though less researched, significantly impacts older workers, leading to lower hiring rates despite potential contributions such as mentorship. Implicit biases against older applicants persist, even in AI-driven hiring processes, highlighting the need for proactive measures to address ageism in recruitment practices.

3.2 Enhancing the hiring experience

In Subsections 2.4 and 2.5, we mentioned that, ideally, the use of AI would make for a better candidate experience. To accomplish this better experience we need to fix some of the leaks in the hiring pipeline. Making the hiring pipeline more transparent, fair and inclusive can be beneficial for both the candidate and the company or its recruiters, as it would make the experience more efficient and pleasant.

From the study by Jayaratne et al. [7], as discussed in Section 2.5, we found that the textual content of open-ended interview questions can be used to infer the personality of candidates. This finding could also be used to form a questionnaire that candidates fill in before they are invited for the first round of interviews. These answers could be analysed to predict how well a candidate would fit the role, which could be used to decide which candidates should be invited to interviews. This saves the hiring companies time and effort by ensuring that the invited candidates are good fits for the role. At the same time, this makes the experience for the candidates better too as unfitting candidates will not have to waste their time interviewing for a role they will likely not get hired for. If this practice were adopted, companies need to be careful regarding data privacy and security as this information is not only valuable to the company itself but also to third parties, as discussed in Subsection 2.3.

3.3 Considerations when using AI technology in hiring

In Section 2.4, we mentioned that the usage of AI comes with concerns about human values, fairness, safety and privacy. To make use of AI technologies responsibly to avoid bias our research pointed out that we need to carefully consider guidelines in the following areas:

- Design of AI systems: AI applications should be designed with the requirement to ensure the participation of all stakeholders, including minorities. An AI application should also be transparent so that users can trace the decisions that the model made to adhere to a person's right to explanation. Additionally, privacy and

security should be well considered and implemented to protect the sensitive data of individuals.

- Choice of (training) data: Training data should be chosen carefully to ensure that the data represents the target domain.
- Testing phase: AI models should be tested thoroughly on bias in their decisions before they are deployed and used.
- Rules and regulations: Teams developing AI models should be aware of rules and regulations to ensure they are developing a model that satisfies the necessary rules.

4 Conclusion

In the introduction, we formulated our research question as follows:

"What fields and factors regarding discrimination in hiring practices have been researched thoroughly and what fields still need to be addressed?".

From the Results in Section 3, we find that there is already a lot of research focusing on the different fields of human and algorithmic bias in recruitment and crowdsourcing. We discovered that in human bias, we can oftentimes find a bias blind spot where individuals are not aware of their own biased thinking but can recognise the same bias in the decisions of their peers. For algorithmic bias, we found that multiple studies focus on the general impact of AI and there are even studies that focus on how to avoid bias in AI to help reduce bias in hiring practices.

In our literature research, we found a paper by Liu et al. [9] that gave advice or guidelines to reduce bias in the hiring process. This advice included considerations about the design of the AI application, the choice of training data and the importance of the testing phase. Additionally, if personal or other sensitive data is used at any point, companies must be careful to protect the privacy of the individual and the security of the data. We also found a study by Hemalatha et al. [6] that highlights that AI should be embraced to extend and improve the hiring process, not to fully replace it.

From the related work and the results, we find the following gaps in research to tackle the issue of human and algorithmic bias in recruitment and crowdsourcing. One future research question is whether adverse working conditions might lead to higher dropout rates among women and minorities in freelancing [4]. This could be explored through an observational study over a longer time. Another challenge is to find the precise influence of social feedback on customers' hiring decisions. This could be investigated through an in-person experiment using a simulated online freelancing platform with real data, allowing for controlled prompts and diverse worker profiles. Lastly, we found that the most researched biases were age, ethnicity, and gender. More research could be done on other factors such as location, nationality, and religion. These future studies could provide deeper insights into the dynamics of freelancing marketplaces and the impact of social feedback on workers and customers.

4.1 Limitations

It is important to highlight the limitations of our paper and it is crucial to recognize and address specific limitations. For example, one such limitation was the exclusion of papers not accessible through the university database. Additionally, subjective judgment was used to decide on the inclusion of certain articles, influencing the formulation of final findings. It is also good to acknowledge the potential oversight of other relevant articles inadvertently missed during our research. Considering the significance of the topic, it's plausible that new and enhanced articles will continue to surface regularly, enriching the current literature.

Acknowledgements

The authors wish to thank our supervisor Dr. Mirela Riveni for her support and guidance.

References

[1] M. Behroozi, S. Shirolkar, T. Barik, and C. Parnin. Debugging hiring: What went right and what went wrong in the technical interview process. In *2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, pages 71–80, 2020.

[2] W. Christl and S. Spiekermann. *Networks of Control: A Report on corporate surveillance, Digital Tracking, Big Data Privacy*. Facultas, 2016.

[3] K. D. . S. M. Hangartner, D. Monitoring hiring discrimination through online recruitment platforms. *Nature*, 589(7843):572–576, January 2021.

[4] A. Hannák, C. Wagner, D. Garcia, A. Mislove, M. Strohmaier, and C. Wilson. Bias in online freelance marketplaces: Evidence from taskrabbit and fiverr. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, page 1914–1933, New York, NY, USA, 2017. Association for Computing Machinery.

[5] C. G. Harris. Age bias: A tremendous challenge for algorithms in the job candidate screening process. In *2022 IEEE International Symposium on Technology and Society (ISTAS)*, volume 1, pages 1–5, 2022.

[6] A. Hemalatha, P. Kumari, N. Nawaz, and V. Gajenderan. Impact of artificial intelligence on recruitment and selection of information technology companies. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, pages 60–66, 2021.

[7] M. Jayaratne and B. Jayatilleke. Predicting personality using answers to open-ended interview questions. *IEEE Access*, 8:115345–115355, 2020.

[8] T. W. Kim and B. R. Routledge. Informational privacy, a right to explanation, and interpretable ai. In *2018 IEEE Symposium on Privacy-Aware Computing (PAC)*, pages 64–74, 2018.

[9] X. M. Liu and D. Murphy. Applying a trustworthy ai framework to mitigate bias and increase workforce gender diversity. In *2022 IEEE International Symposium on Technology and Society (ISTAS)*, volume 1, pages 1–5, 2022.

[10] S. Nabi. Comparative analysis of ai vs human based hiring process: A survey. In *2023 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 432–437, 2023.

[11] A. Peng, B. Nushi, E. Kiciman, K. Inkpen, S. Siddharth, and E. Kamar. What you see is what you get? the impact of representation criteria on human bias in hiring. In *Proceedings of the 7th AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2019)*, pages 125–134. AAAI, 2019.

[12] P. A. Riach and J. Rich. An experimental investigation of sexual discrimination in hiring in the english labor market. *The B.E. Journal of Economic Analysis Policy*, 6(2):000010220215380637141 6, 2006.

[13] T. Sühr, S. Hilgard, and H. Lakkaraju. Does fair ranking improve minority outcomes? understanding the interplay of human and algorithmic biases in online hiring. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '21, page 989–999, New York, NY, USA, 2021. Association for Computing Machinery.

[14] O. Thomas and O. Reimann. The bias blind spot among hr employees in hiring decisions. *GERMAN JOURNAL OF HUMAN RESOURCE MANAGEMENT-ZEITSCHRIFT FUR PERSONALFORSCHUNG*, 37(1):5–22, FEB 2023.

[15] R. F. West, R. J. Meserve, and K. E. Stanovich. Cognitive sophistication does not attenuate the bias blind spot. *JOURNAL OF PERSONALITY AND SOCIAL PSYCHOLOGY*, 103(3):506–519, SEP 2012.

[16] X. Zhao and R. Young. Workplace discrimination in software engineering: Where we stand today, 2023.

Faint Object Detection Methods in Optical Astronomy

Tom Couperus and Theo Krijgsheld

Abstract—The detection of low surface brightness objects is a difficult problem in optical astronomy. Over the years, multiple tools have been developed to better detect and classify these faint objects. Some of these tools exploit knowledge of the noise characteristics found in the observation data, while other tools apply deep learning to the problem. This first type of approach has been studied and compared to others of its type, such as in Haigh et al. 2021.
Our research aims to provide an overview of both types of approaches. We cover more traditional, characteristics-based approaches through tools such as SExtractor, ProFound, NoiseChisel, and MTObjects, but also deep learning approaches such as support vector machines (SVMs), convolutional neural networks (CNNs), and Mask R-CNN. We compare the approaches and analyse their strengths and weaknesses. We base our comparison on whether each method can detect objects as well as classify them. After compiling this information, we suggest new validation methods for these approaches and suggest ways for improvement.
We summarize each of these tools and the dataset used for their testing. To do this, we rely on existing papers that cover the tools and techniques and compare those findings and the datasets used in their experiments. Our comparison is based on the metrics of accuracy, precision, completeness, and speed.
Our comparison shows that deep learning techniques are starting to approach the classification power of characteristics-based approaches on most metrics we compare. However, the deep learning methods still require noise-based methods to detect any objects. The highest precision score, and thus type I error minimization, was obtained by a characteristics-based method. We suggest that a combination of the two approaches could lead to progress in the field.

Index Terms—Optical astronomy, low surface brightness objects, image processing, deep learning, convolutional neural networks, survey.

1 Introduction

Learning about the universe requires us to analyze astronomical surveys. With the evolution of techniques to observe the universe, these surveys can grow incredibly large. Recent projects aim to produce large amounts of data. The Large Synoptic Survey Telescope [10] aims to produce 15 TB of data per night, in the form of raw imaging data. This amount of data is unfeasible to analyze by hand and, therefore, there is a need for automated methods.

Existing automated methods have proven inadequate in recognizing Low Surface Brightness Galaxies (LSBGs) due to their similarity to other low surface brightness objects. These artifacts can be faint, compact objects blended in the diffuse light from nearby bright stars, giant elliptical galaxies, or bright regions of galactic cirrus [16]. In a recent study that used an automated analysis pipeline [5], much data had to be manually inspected due to a large number of incorrect classifications in the research. About 50% of the objects classified as LSBG were actually another low surface brightness object.

Because of this, new methods have been developed to increase the productivity and accuracy in the analysis of astronomical data. Older methods like SExtractor, ProFound, NoiseChisel, and MTObjects use knowledge of noise characteristics of the optical data. Newer methods rely on deep learning, such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Mask R-CNN. To get a better understanding of these new methods, we will summarize how they work and how they perform. This may also reveal directions for future research. We will analyze and compare their strengths and weaknesses in search of possible improvements, and aid future researchers in choosing the right tool for a specific circumstance.

In Section 2 we explain those tools that use knowledge of noise characteristics, while in Section 3 we explain tools that utilise deep learning. We will compare these two approaches in Section 4. Lastly, in Section 5 we will conclude our research, and in Section 6 we give suggestions for future research.

- *Tom Couperus is with University of Groningen, E-mail: t.couperus@student.rug.nl.*
- *Theo Krijgsheld is with University of Groningen, E-mail: t.krijgsheld@student.rug.nl.*

2 Summary and Analysis of Characteristics-based Approaches

In this section, we summarize four astronomical segmentation tools that exploit knowledge of photon noise characteristics to detect and classify LSBGs. These methods were all compared in [6], using the same dataset and testing procedure.

The data and testing procedure is explained in Section 2.1, while Sections 2.2 through 2.5 summarize the four tools.

2.1 Data and Testing

In the previous comparison study [6], two types of data were used. The first type was a simulated dataset, which was used to accurately determine the performance on samples with a known ground truth. The second type of data is gathered from multiple real astronomical surveys. With this dataset, the authors researched the behaviour of the methods with artifacts which are harder to simulate.

The synthetic dataset consists of 10 generated images, each containing approximately 1500 stars, 400 cluster galaxies, and 50 background galaxies. The images are based on the data of the Fornax Deep Survey [9, 18]. Setting the ground truth of such images can be challenging, as there is no clear boundary of astronomical objects. The light they emit only fades to an insignificant value, when compared to noise and the background. For the selection of the ground truth, a threshold is used. Below this threshold, the light is deemed to be undetectable. There is also the problem that two light sources can overlap. In this case, one pixel contains light of two sources and the pixel is assigned to the source which contributes the greatest part of the total brightness of the pixel. For more detailed information on the generation of the data and the selection of the ground truth, please refer to [6].

The real data is selected from various astronomical surveys but is limited to optical wavelengths to test the limits of these methods. The first is the Fornax Deep Survey again. Since the synthetic data is also based on this survey, the optimized parameters of the synthetic tests are expected to perform well on this data too. Furthermore, two other real astronomical surveys, IAC Stripe 82 Legacy Project and Hubble Ultra Deep Field, were involved in researching the consistency of the parameters in different conditions. An example image is displayed in Figure 1.

The study reported the performance of each tool using detection rates. For these rates, it may be the case that a detected object covers more than one ground truth object or vice versa, a ground truth object

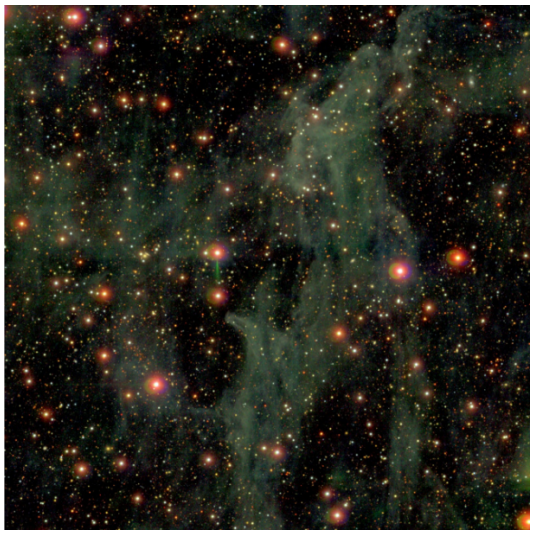


Fig. 1: Test image in the research of the characteristics-based approaches of the IAC Stripe 82 Legacy Project. Image taken from [6].

covers more than one detected object. Therefore a one-to-one mapping was created which leads to algorithms being rewarded when they can correctly distinguish different sources. The authors implemented this mapping by taking the highest valued pixel of the detected object and matching it to the highest intensity ground truth, which the area covers.

The detection rates were reported using three metrics. The precision is "the proportion of segments that can be matched to real objects" [6]. The completeness is "the proportion of objects that are detected" [6]. These metrics are calculated as follows: $precision = \frac{TP}{TP+FP}$ and $completeness = \frac{TP}{TP+FN}$. TP is the number of true positives or the amount of light sources that were correctly classified as light sources. FP and FN are the amount of false positives and false negatives, respectively. FN is the number of light sources that were not detected and FP is the number of light sources that were detected while there was nothing to be detected. The final metric is the F-score, which is calculated using the precision and completeness:

$$\text{F-score} = \frac{2 \times precision \times completeness}{precision + completeness} \quad (1)$$

2.2 SourceExtractor

SExtractor (SourceExtractor) was introduced in 1996 and its primary goal is to separate stars from galaxies [2]. It was designed to be fast and robust to process large amounts of survey data.

The tool first estimates and subtracts the background. To do this, the image is divided into tiles. For each tile, a histogram is made and its values are clipped in a range of 6 standard deviations around the median. Depending on the amount of change in the distribution, the background value for each tile is calculated using the mean or the mode of the tile. A binary interpolation on the tiles results in the final background map.

Objects are then detected using a multiple threshold method. This method assigns pixels with similar intensities to an object. It also may lead to cluttering objects together, so a second pass is needed to separate these objects. This is done by thresholding each set of connected pixels again on an exponential scale, resulting in a light distribution which is stored in a tree-like structure. The final deblending is done by moving down the tree, from the branches to the trunk, and determining whether to separate pixels or keep them together at each node, based on their intensity difference.

SExtractor performed decently with F-scores ranging from 0.75 to 0.79 [6]. It showed little variance when the results were grouped by the images they were tested on, which suggests that this method is limited by the structure of the test image, rather than the parameters. When analyzing the precision and completeness separately, the authors observed that the precision was quite high, with all values higher than 0.93. The completeness was lower, with values all below 0.66. However, to completely analyze the performances of this method, it is important to note that a lot of the objects were very faint and not even detectable by the human eye. Some of the faintest objects may be impossible to detect with any automated method. Therefore the completeness seems low, but it will consistently have relatively low values when compared to precision for the other noise-based methods.

As mentioned at the start of this section, SExtractor was designed for speed and it proved to have this feature. However, it is highly dependent on the selected parameters. When optimized for F-score there was a low amount of spread across the speeds of SExtractor. However, if the parameters were changed, the algorithm showed a lot of varying speeds and it resulted in a slower average.

2.3 ProFound

Similar to SExtractor, ProFound was designed to detect sources, like stars and galaxies, from noisy astronomical images [14]. It is also able to generate segmentation maps, which leads to a more accurate assignment of pixels to their source object.

ProFound creates an estimation of the background using the same approach as SExtractor. However, instead of using multiple thresholds when segmenting the pixels into objects, ProFound uses a watershed-like algorithm. The algorithm first flags pixels which have a higher intensity than the requested threshold. Then it takes the brightest flagged pixel and searches pixels around this flagged pixel. If a pixel is found with a lower intensity than the segment, the pixel is added to the segment, and if the pixel is above the segment (and some added tolerance) the pixel is not added to the segment. When there are no more pixels to add, the segment is completed and the growing process is terminated. After this, the background is re-estimated and the final segmentation map is produced by repeated dilation of the segments.

The F-scores for ProFound range from 0.77 to 0.81, the completeness ranges from 0.63 to 0.69 and the precision is all above 0.95 [6]. In contrast to SExtractor, the F-scores of ProFound were more spread out when grouped by test image. However, when grouped by the image on which the algorithm was optimized, this spread decreases a bit. This may indicate that this method is limited by the structure of the image which is used to optimize its parameters.

ProFound consistently showed large processing times, since it was designed to be robust instead of fast [14]. The low speeds can be explained by the fact that this method involves writing temporary data to the disk. In the previous comparison study in [6], ProFound was observed to be too slow for some of the more lengthy experiments they performed. However, the authors also mentioned that the developers of ProFound were rewriting parts of the algorithm, so a newer version might show more promising speeds.

2.4 NoiseChisel

The main goal of NoiseChisel [1] is to find nebulous objects in images, such as irregular galaxies and their structure. It is a non-parametric method, which means that it does not include a regression analysis

NoiseChisel separates the background from the objects using one threshold which is just below the estimated background threshold. It then applies two binary morphological filters: one erosion and one opening. This creates the initial detection map. Additional morphological operations are carried out on the background and the objects separately. Afterwards, the connected components are filtered by their signal-to-noise ratio.

The image is then further segmented by finding local maxima and using a watershed-like approach to grow the maxima into segments. It also removes the regions that have a too low signal-to-noise ratio.

NoiseChisel performed consistently higher than SExtractor, with F-scores ranging from 0.76 to 0.80 [6]. This increase is also visible in the precision, which was above 0.95. Furthermore, it performed best in estimating the background values. This is however at a cost, as it occasionally divides detected light into seemingly random segments.

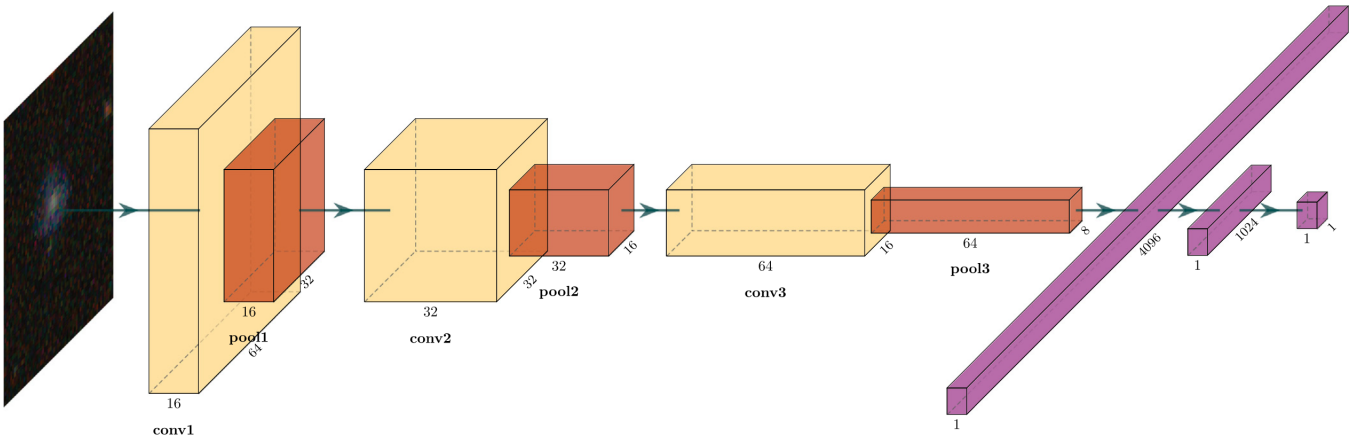


Fig. 2: Architecture of DeepShadows CNN. The network consists of three (yellow) convolutional blocks, each followed by a (red) max pooling layer. Two (purple) fully connected layers process the flattened output of the last pooling layer. Image taken from [16].

Metric	SE	PF	NC	MTO
Completeness	61 – 66%	63 – 69%	62 – 66%	66 – 69%
Precision	> 93%	> 95%	> 95%	> 98%
F-score	75.5%	77.9%	76.5%	79.9%

Table 1: Metrics of comparison study of characteristics-based approaches with parameter optimisation [6]. SE = SExtractor, PF = ProFound, NC = NoiseChisel, and MTO = MTObjects.

The speed of NoiseChisel was moderate. Similar to SExtractor its speed was dependent on the parameters and some parameters lead to less spread speeds than others. Furthermore, the method was slowed down by a similar issue as ProFound, it required large amounts of memory and started writing to the disk.

2.5 MTObjects

MTObjects was designed with SExtractor as reference [17]. It uses a similar approach to subtract the background. However when detecting objects, instead of using a small number of fixed thresholds, MTObjects uses a max-tree, leading to the name 'Max-Tree Objects'.

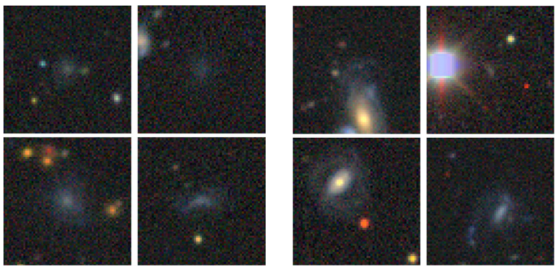
Any gray-scale image can be represented using a max-tree. In the max-tree, the root is the entire image and the leaves are the local maxima. Each node is investigated and a few statistical tests are applied on it to determine which of them are significant when compared to the background. A node is marked as a separate object if it has no parent with greater intensity or if its parent has a child with greater intensity.

MTObjects resulted in the highest F-scores of the classical method, ranging from 0.79 to 0.82 [6]. Its precision values were all above 0.98 and the completeness ranged from 0.66 to 0.69. Furthermore, it showed a low spread of values in all metrics. A weakness of MTObjects is the edges of objects, as they looked jagged, however, it performed very well in fainter regions.

MTObjects showed very consistent speeds. Using different parameters did not significantly change the speed of the algorithm. Unfortunately, it only performed at mediocre speeds. However, the version of MTObjects that was used was not parallelized yet, and there has already been research into parallelizing a max tree [13] so this method might be faster and more promising to use in the future.

3 Summary and Analysis of Deep Learning Approaches

Next, we summarize and analyze four deep learning approaches to detecting and classifying LSBGs from artifacts with similar characteristics. Three of these approaches were performed in the DeepShadows study [16] and were tested on the same dataset. The fourth approach was performed in a separate study [12], using a different set of data.



(a) Example LSBGs (b) Example artifacts

Fig. 3: Example images of the dataset used by the DeepShadows study. Images taken from [16].

Section 3.1 goes over the three methods tested in the DeepShadows study, while Section 3.2 explains the fourth approach.

3.1 DeepShadows

The study that conceived DeepShadows investigated the potential of classical support vector machines (SVM) [4], the random forest (RF) model [3, 8], and the titular convolutional neural network (CNN) [11] as classifiers of LSBGs in astronomical images.

The main dataset used in the study is from the Dark Energy Survey's (DES) first three years. The DES dataset for the study was filtered down to more than 40,000 samples in four main steps [15]. The first of these steps involved sectioning cuts by SExtractor (see Section 2.2), bringing the total down to ∼420,000 objects. The second step performs a classification by an SVM trained on SExtractor parameters and ∼8,000 objects, further reducing the sample size. In the third and fourth steps, manual inspection of more than 40,000 objects positively classified as LSBGs and further corrections created a high-purity dataset of 23,790 LSBGs.

For the testing of the following methods, the final sample set consists of 20,000 visually verified positive LSBG objects (see Figure 3a) and an equal amount of visually verified artifact objects (see Figure 3b), all chosen randomly from the third step in the filter process. These 40,000 objects are split into a training, validation, and testing set of 30,000, 5,000, and 5,000 objects, respectively.

The classical machine learning methods and the CNN do not operate on the same data representation of these objects. The SVM and RF use the properties of each object derived from SExtractor, while DeepShadows operates on image cutouts of each object. For DeepShadows, each image cutout is 64x64 pixels in 3 channels. The channels of the image represent the g, r, and z astronomical bands. Each image represents 30″ x 30″ of the sky, centered on the candidate

Metric	SVM	RF	CNN
Accuracy	81.9%	79.7%	92.0%
Completeness	86.7%	80.4%	94.4%
Precision	79.6%	79.7%	90.3%
F-score	83.0%	80.0%	92.3%
AUC	0.894	0.872	0.974

Table 2: Metrics of DeepShadows study [16].

Metric	Direct transfer	With fine-tuning
Accuracy	82.1%	87.6%
Completeness	84.9%	84.1%
Precision	78.6%	93.2%
F-score	81.6%	88.4%

Table 3: Metrics of the potential of transfer learning of DeepShadows CNN [16].

object. The CNN's architecture is shown in Figure 2.

Each of the three tested methods outputs a probability of an object being an LSBG. This probability is then compared against a threshold probability $P = 0.5$ to determine the final predicted label. Any value above P is labelled as LSBG (positive), and below P as an artifact (negative). The predicted label is compared against the true label of the object, giving the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) counts for each method.

The study then evaluates each method by analysing statistics derived from the ratios of these four statistics. Specifically, the ratios of $accuracy = \frac{TP+TN}{TP+TN+FP+FN}$, $completeness$, and $precision$ are measured (see Section 2.1). The receiver operating characteristic (ROC) curve plots the completeness against the false positive rate $FPR = \frac{FP}{FP+TN}$ at different thresholds in a visual way. The area under this curve (AUC) is used as another metric in the study.

The results in Table 2 were obtained from the validation set after training. It shows that the CNN is a significantly more powerful classifier, compared to the SVM and RF methods.

Using Gradient-weighted Class Activation Mapping (Grad-CAM), the authors found that images with off-centered bright areas were all classified as artifacts, whether they actually were artifacts or not. Similarly, images with light sources in the center were classified as LSBGs, correctly or not. The authors note that the false LSBGs seem to be actual galaxies, which were labelled artifacts due to human caution, and DeepShadows corrected those cases.

Label noise was also considered in the study. Inverting the labels of 1%, 5%, 10%, and 33% of the training set, retraining, and evaluating on the test set, it was found that low percentages of noise had little to no influence on the performance. Only at 33% noise did a decrease in accuracy become noticeable, indicating that the CNN is robust to noise.

Lastly, the authors investigated the ability to use the trained DeepShadows network on a different image dataset. The dataset used is the HSC SSP dataset [5], which has different biases.

The potential of transfer learning was evaluated in two ways:

1. Directly applying the DES-trained DeepShadows model on the HSC SSP dataset.
2. Using a quarter of the new dataset to further train the DES-trained model before applying it to the remainder of the new dataset.

As Table 3 shows, good performance can be reached with direct transfer learning, and even better performance by using a small sample of the new dataset to further train the model.

3.2 Mask R-CNN

The second deep learning approach to detecting LSBGs that we consider uses the Mask R-CNN framework [7]. The purpose of this approach is to "detect large, faint LSBGs that are missed by conventional algorithms" [12].

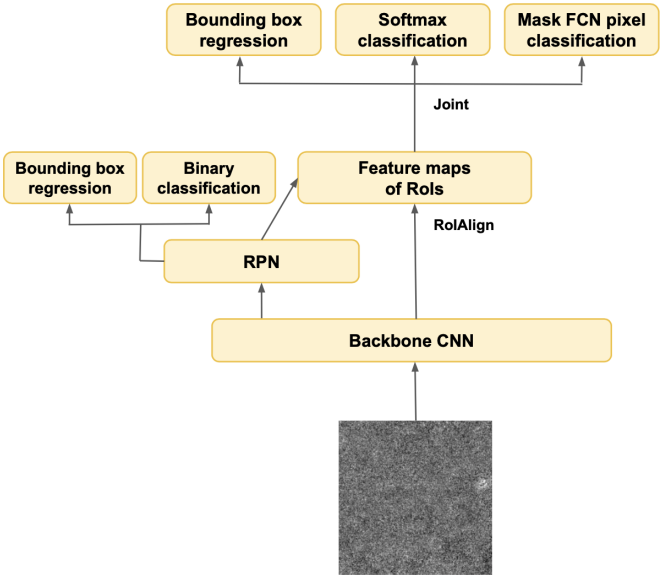


Fig. 4: Mask R-CNN structure. Image taken from [12].

The Mask R-CNN model (see Figure 4) is a segmentation and object detection algorithm, and works as follows: An image is fed into a pre-trained network that outputs a feature map. The study used ResNet as the network for this. This feature map is used to detect regions of interest (RoI) with a dedicated CNN. The RoIs and the feature map are used together, after more processing, to classify the object with a softmax classifier, to obtain bounding boxes of the objects, and to obtain a pixel-level classification with a fully convolutional network.

The study used images from the Deep Energy Survey's (DES) second data release. These images are $10,000$ x $10,000$ pixels in size. The authors randomly chose 94 of these images and injected 16 simulated LSBGs into each image, such that there was one LSBG per 2500 x 2500 pixels segment.

The edges of light sources can contaminate the data, so to remove this contamination, the authors added a separate class for large objects. They defined large objects as those masked areas exceeding $10,000$ pixels in angular size. The model was trained on 1034 simulated LSBGs, and tested with 282.

The authors judged their model qualitatively, and, once satisfied with the test performance, used the model on the whole DES dataset of $10,169$ images. They report that $13,336$ objects were found and classified as LSBGs, of which 41 were visually judged to be LSBGs with any amount of confidence. The locations of other detected objects were all found to have a high visual correlation with known galactic cirrus, light scattered by space dust, except for a few indicators of galaxy mergers.

The Mask R-CNN study does not report the number of objects detected in other classes, nor does it calculate metrics such as accuracy, precision, or completeness. However, we can set the amount of true positives equal to the amount of correctly classified LSBGs, and false positives to artifacts classified as LSBGs, as these values were reported. This allows us to calculate only the precision of correctly finding LSBGs, which results in $precision = 41/13,336 = 0.3\%$.

4 Comparison

We now compare the performance of characteristics-based detection and classification methods to deep learning approaches. We compare the statistics of accuracy, precision, and completeness of each method, as well as their speed. The speed of each method differs depending on the type of approach. For the characteristics-based approaches the speed is the processing speed, but for deep learning approaches the training time is also factored in.

It is important to note that the methods researched in the DeepShad-

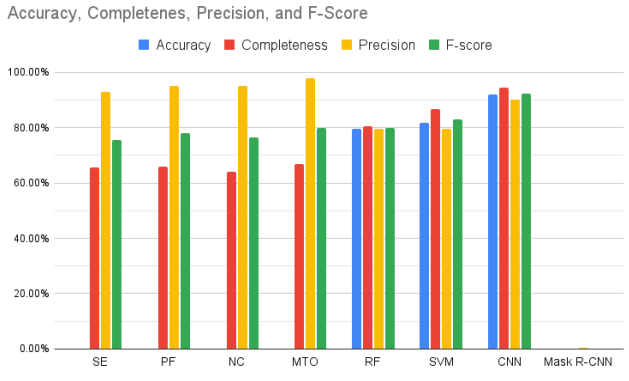


Fig. 5: Performances of the tools reported as percentages. Where ranges were given, the mean of this range was used.

ows study (Section 3.1) are only classification methods, compared to the detection and classification tests performed by the other studies (Sections 2 and 3.2). The three methods proposed by DeepShadows require other tools such as SExtractor to detect any objects, whereupon the RF, SVM, and CNN can decide whether an object is an LSBG or an artifact. Therefore, we must keep in mind that the classification-only methods are limited by the performance of the detection methods.

Another important aspect is that some studies merely reported on some metrics that their results were inadequate, while others did not report these metrics at all. Where possible, we calculated the missing metrics from the results that were provided in Sections 2 and 3. We plotted the available accuracy, precision, completeness, and F-score of each method in Figure 5.

Finally, it should be noted that our comparison is performed on non-uniform testing data. As explained in Section 2 and Section 3, each study used a different dataset for comparison. Those in Sections 2 and 3.2 used simulated datasets with generated ground truths, while the study covered in Section 3.1 used a dataset that consists of real objects detected by SExtractor and further refined by manual inspection to achieve an accurate ground truth.

Immediately evident from Figure 5 is the wide difference between completeness and precision for the characteristics-based methods, with precision being the higher of the two.

Precision is a measure of how well a method positively classifies detected objects. Classifying negatives as positives lowers the precision, as does misclassifying positives as negatives. A high precision minimizes the type I error rate and reduces the need for visual inspections to filter out false positives. For the characteristics-based methods, high precision is especially good, as the images these four methods were optimized and tested on were designed to test the limits of each method. MTObjects performs the best of the detection methods, in that scenario, with a precision of $> 98\%$.

This limit-testing of the noise-based approaches showed itself in their comparatively low completeness. Completeness minimizes type II errors. A high value indicates that all known positive objects were correctly classified, ignoring possible negatives classified as positive. The low completeness of the noise-based methods indicates that positive objects were not correctly classified, and is likely caused by the incredibly faint, simulated objects that the methods couldn't detect.

The other method that performs detection of objects is the Mask R-CNN. The study did not publish many statistics, but we were able to derive a precision score from the published data. It was, as Figure 5 shows, incredibly low. This precision score is calculated as the rate of correctly classifying LSBGs, which was verified by manual classification of objects detected by the framework. A completeness score is unavailable, as no ground truth was available for the dataset used. Compared to noise-based detection methods, there is room for improvement among the deep learning detection approaches.

In contrast to the noise-based methods, the deep learning classification approaches have a much smaller gap between precision and completeness scores. The completeness of the deep learning approaches is significantly higher, indicating that more of the objects were correctly classified. For the SVM and RF methods, the precision is lower than any of the optimized characteristics-based methods, meaning that despite removing the detection portion of the task, these methods are not better than noise-based approaches. The DeepShadows CNN, however, approaches the level of precision of the characteristics-based methods, with better completeness, but is limited by the fact that it can only classify objects that are already detected by segmentation tools such as SExtractor.

Additionally, this improvement in completeness is not without compromises. A CNN requires significant amounts of time to train on a single sufficiently large dataset, compared to a program such as SExtractor, which is not bound to a specific dataset and can be run without training time on a single image.

However, this flexibility of noise-based methods is also not without cost. The authors of [6] have found that the speed of all characteristics-based methods is dependent on parameter choice, except for MTObjects. Additionally, they found that the speed of the characteristics-based methods was insufficient for use in large-scale, automated segmentation. The flexibility is also hampered by the study's finding that the optimisation method was limited for all tools, except for MTObjects again. This was the only tool that reliably found the global optimum of parameters. The best selection of parameters is believed to differ from dataset to dataset, and in some cases was found to differ from image to image within the same dataset.

The DeepShadows CNN showed more promise in the transfer learning aspect. This was initially investigated as a way to reduce the need for generating large, visually confirmed datasets for training future CNNs, but also serves as a decent comparison to the noise-based methods. Overall, the DeepShadows CNN showed good results with fine-tuning of a trained CNN. A decrease in completeness and an increase in precision were observed, showing deep learning's potential for transfer learning.

Apart from completeness, precision, and speed, another aspect we would have liked to compare is accuracy. It is a more generic metric than precision or completeness, as accuracy measures the ratio of correct classifications of detected objects, regardless of class. However, we only know the accuracy scores of the three deep learning classification methods, and we have no information on the detection methods' accuracy. The researchers of the original comparison study on characteristics-based methods [6] only stated that "no tool has sufficient speed and accuracy to be well suited to large-scale automated segmentation in its current form". This commentary is still useful information, but the lack of reported accuracy values limits the value of any comparisons we would make.

The final metric to compare is the F-score of each method. The F-score, as discussed in Section 2.1, is a ratio of precision and completeness. A high F-score therefore minimizes both type I and type II errors. The DeepShadows CNN scores the best on this metric, due to its high precision and completeness scores. However, as we have stated before, this method is limited by the detection performance of other methods such as SExtractor. Of the detection methods, MTObjects has the highest F-Score, since it also has the highest completeness and precision.

5 Discussion and Conclusion

In conclusion, we have summarized multiple methods for detecting and classifying low surface brightness galaxies. We compared four methods that detect and classify these celestial objects from a photon noise characteristics-based standpoint, as well as one method that uses deep learning. We have also compared the classifying aspect of these five methods to three more approaches that apply deep learning to the problem.

We compared these methods on the metrics of accuracy, precision, completeness, and speed. However, our comparison is based on studies that used different datasets to establish these metrics, and three of

the deep learning-based methods rely on detection tools to supply their data, instead of detecting objects themselves.

The characteristics-based detection and classification tool MTObjects has the highest precision, at 98%. In fact, the methods that use knowledge of photon noise characteristics had the best precision, compared to deep learning methods. This is promising, as minimizing type I errors reduces the need for visual verification of results. The drawback of the characteristics-based methods, even after parameter optimisation, is their low completeness, leading to a portion of LSBGs getting classified as artifacts. However, these metrics were obtained by using images designed to test their limits, instead of more conventional images. With more realistic images the completeness might perform better, but parameter selection is still a matter of trial and error without scientific justification, according to [6].

Our comparison showed that the DeepShadows CNN performed the best on these metrics overall. However, this overall high performance does not mean that it correctly detected more LSBGs than the noise-based methods, as it cannot detect objects from a larger image. Instead, the network was designed to classify the object in an already cropped image. The dataset used to train and test DeepShadows was compiled using SExtractor and manual verification of SExtractor's findings.

Additionally, the CNN has some limitations in the form of its speed. A CNN requires time to be trained on a specific dataset. Yet this can be mitigated if the trained network can be applied to other datasets via transfer learning. The DeepShadows study has shown that their CNN has good results with this. Nonetheless, this retraining time cannot be ignored.

The ability to transfer learned optimal parameters of the noise-based methods is limited. The parameter selections that were obtained from the optimisation method were found to be subjective to the dataset used, and in some cases subjective to the image used. The speed of characteristics-based methods when used in large-scale automated segmentation is also lacking.

The Mask R-CNN framework that was applied to the problem of detecting faint objects had poor results for detecting LSBGs. The framework's other detected objects, however, were found to have a high correlation with the locations of known galactic cirrus, indicating that the framework has the potential to detect these artifacts, instead of LSBGs.

Lastly, we were unable to compare all methods on accuracy, as only the three deep learning classification methods reported this metric. All detection methods did not report their accuracy.

Since the noise-based tools performed with the best precision on detection tasks that pushed their limits, we recommend the use of MTObjects to detect objects in a conventional setting. Should the low completeness of the limit-testing persist during conventional use, perhaps less strict parameters could be used to generate more locations, which can be eliminated by a deep learning classification method such as DeepShadows.

6 Future Work

As the eight methods we compared were evaluated on three different datasets, we see future work do the comparison of all tools on the same dataset. This future study would also compare the methods on all metrics, instead of the limited comparison we were able to make with missing metrics.

Another avenue of future research could be a multi-stage detection framework. As we recommend in our conclusion, the combination of a noise-based detection tool operating on loose parameters to generate many points of interest with a deep learning-based classification tool could have potential.

Acknowledgements

We give our thanks to Björn Schönrock and Hayo Riem for their aid in structuring our paper and pointing out areas where clarification was needed. The invaluable input of Michael Wilkinson is also greatly appreciated as it significantly improved the value of our comparison.

References

[1] M. Akhlaghi and T. Ichikawa. Noise-based detection and segmentation of nebulous objects. *The Astrophysical Journal Supplement Series*, 220(1):1, Aug 2015.

[2] E. Bertin and S. Arnouts. SExtractor: Software for source extraction. *astronomy and astrophysics supplement series*, 117:393–404, Jun 1996.

[3] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[4] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273 – 297, 1995.

[5] J. P. Greco, J. E. Greene, M. A. Strauss, et al. Illuminating Low Surface Brightness Galaxies with the Hyper Suprime-Cam Survey. *The Astrophysical Journal*, 857(2):104, Apr 2018.

[6] C. Haigh, N. Chamba, A. Venhola, et al. Optimising and comparing source-extraction tools using objective segmentation quality criteria. *Astronomy and Astrophysics*, 645:A107, 2021.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN, 2018.

[8] T. K. Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[9] E. Iodice, M. Capaccioli, A. Grado, et al. The Fornax Deep Survey with VST. I. The Extended and Diffuse Stellar Halo of NGC 1399 out to 192 kpc. *The Atstrophysical Journal*, 820(1):42, Mar 2016.

[10] Ž. Ivezić, S. M. Kahn, J. A. Tyson, et al. LSST: From Science Drivers to Reference Design and Anticipated Data Products. , 873(2):111, Mar 2019.

[11] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] C. Levy, A. Ciprijanovic, A. Drlica-Wagner, et al. Detecting Low Surface Brightness Galaxies with Mask R-CNN. In *Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021)*, Jan 2021.

[13] U. Moschini, A. Meijster, and M. H. F. Wilkinson. A hybrid shared-memory parallel max-tree algorithm for extreme dynamic-range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):513–526, 2018.

[14] A. S. G. Robotham, L. J. M. Davies, S. P. Driver, et al. ProFound: Source Extraction and Application to Modern Survey Data. *Monthly Notices of the Royal Astronomical Society*, 476(3):3137–3159, Feb 2018.

[15] D. Tanoglidis, A. Drlica-Wagner, K. Wei, et al. Shadows in the Dark: Low-surface-brightness Galaxies Discovered in the Dark Energy Survey. *The Astrophysical Journal Supplement Series*, 252(2):18, Jan 2021.

[16] D. Tanoglidis, A. Ćiprijanović, and A. Drlica-Wagner. DeepShadows: Separating low surface brightness galaxies from artifacts using deep learning. *Astronomy and Computing*, 35:100469, 2021.

[17] P. Teeninga, U. Moschini, S. Trager, and M. Wilkinson. Statistical attribute filtering to detect faint extended astronomical sources. *Mathematical Morphology - Theory and Applications*, 1(1):100–115, Jan 2016.

[18] A. Venhola, R. Peletier, E. Laurikainen, et al. The Fornax Deep Survey with the VST. IV. A size and magnitude limited catalog of dwarf galaxies in the area of the Fornax cluster. *Astronomy & Astrophysics*, 620:A165, Dec 2018.

Object Detection for Vast Radio Astronomy Surveys

Chris van Wezel

Abstract— As in any other research field the data collection of Vast Radio Astronomy is affected by the non-stop development of better processors and denser, faster storage mediums. The effect is shown in the large amounts of data gathered during a survey and the complexity and massive size of developed survey equipment. With this development, a demand arises for fast, reliable and scalable processing of this data. To address the demand and process the data, several challenges have to be met. The most prominent is the large size of the data and another challenge is the complex types of artefacts within the data. Multiple groups have started to develop methods to address this demand with various results, strengths and weaknesses. This research will focus on reviewing methods of processing data generated from vast radio astronomy surveys used to identify objects. The focus will lie on every type of method, from more traditional to deep learning, where not only the accuracy of the methods will be considered but also how it deals with newer challenges like the increased data complexity and the large volume of data. Using the results an inventory of strategies to solve the new challenges is compiled. This research will judge different methods on their ability to detect objects. To achieve this for each method their accuracy is considered and parallels between methods are discussed where needed. Also important for the performance of the different methods is the platform they are run on. As a secondary goal, we also look at a paper suggesting three different platforms for running the methods. In the end, we conclude that a hybrid version of a CNN and a CNN focused pipeline performs best in source finding and should run most cost and performance efficient on an AWS cloud system.

Index Terms—Radio astronomy, object detection, machine learning, big data, data complexity, .

1 Introduction

Approximately 400,000 years after the big bang the universe entered a period called the dark ages. At this time the universe was devoid of light and the primary source of emitted signals was from hydrogen. Today we know very little about this time due to the expansion of the universe. Signals that were 21cm are now stretched to $\tilde{2}$m due to the Doppler effect, making the signals very faint and difficult to measure.

There have been various projects that aim to detect these kinds of low frequency radio signals. One of the most ambitious currently being build and scheduled to go online in 2027 is the Square Kilometer Array, SKA. It is a global effort that aims to operate over a wide range of frequencies, being able to survey a large part of the sky faster, and have a superior sensitivity in comparison with previously build arrays. Naturally, this comes with new challenges in particular the amount of data that it will generate which is approximately $\tilde{1}$4 exabytes of data per day.

In addition to the massive amounts of data generated the data itself is more complex due to how the radio signals are measured. Signals arrive at different angles all at the same time and there is no way to filter them like with optical astronomy which only deals with photo noise. Signals from one angle can be calculated using some simple geometry, however the signals from different angles act as noise. Furthermore, with the increase in sensitivity higher resolution images can be generated which will expose more complex anomalies previously hidden.

Detecting objects from this set of data is thus not simply to scale the algorithms. Instead, it needs to be carefully considered how algorithms can be scaled and what technical challenges lie ahead.

This paper will survey the methods and aspects of processing the data to give an overview of the different solutions available and discuss their advantages and disadvantages. The outcome should be a better understanding of how to overcome the new obstacles.

We begin by describing the background regarding software and methods used in the competition in Section 2. In Section 3 we will look at proposed frameworks to process the SKA-Scala data efficiently and cheap. After that in Section 4 we will look at the results of the each of the methods and the frameworks and discuss them. In Section 5 we will give our concluding and finally, Section 6 is about Future work.

• *Chris van Wezel is with the Faculty of Science and Engineering, University of Groningen, E-mail: c.s.van.wezel@student.rug.nl.*

2 Background

In this section we will look at several methods that exist to handle the large volume of data created by the SKA. Among these are the algorithms created or used by three teams from the SKA science data challenge and two other methods. SoFiA and a range of CNN implementations are the methods used by the teams. The two other methods are two different implementations of forms of component trees.

2.1 SoFiA

SoFiA is a tool specifically designed for the detection and parametrization of sources in 3D spectral line data sets [12]. Since its creation a second updated version has been released [14]. Many of the teams in the SKA science data challenge used SoFiA for part of their pipeline but, the most notable team, the team that used SoFiA for almost their full pipeline and came in third, was Team Sofia [9]. SoFiA is a modular tool with a set pipeline that allows the user to modify the settings of each module in the pipeline or skip modules completely. Some of these modifications to the pipeline are: adding filters or weights for the input cube, merging detected voxels to identify sources and measuring source parameters. An overview of the pipeline is shown in Figure 1. The software is written in C++ and Python and uses several external libraries. As long as these are available the software can run on any Unix or Linux based system. Besides a simple commandline the tool also contains a dedicated graphical user interface.

For the SKA science data challenge Team Sofia divided the full cube into smaller regions and gave these to 80 instances of SoFiA running in parallel. The 80 instances together took a merely 2h to run. Some of the settings used by Team Sofia were: flagging of bright continuum sources < 7 mJy and noise normalization in each spectral channel. During testing on the development cube the team was able to improve the reliability to an implied 94.2 percent by removing the detections within the generated source catalogue that fell outside a specifically tuned bound.

For an elaborate explanation of the settings used and the improvements made to the pipeline by Team Sofia see Hartley et al. [9].

2.2 Convolutional Neural Network

convolutional neural network (CNN) is a deep learning algorithm that tries to mimic a neural network with the use of one or more convolutional layers, each layer has a set of trainable parameters based on the size of the layer [8]. Through these trainable layers it is possible to learn a CNN to do all different kinds of data manipulation. A CNN can for example be learned to create an image from a random collection of

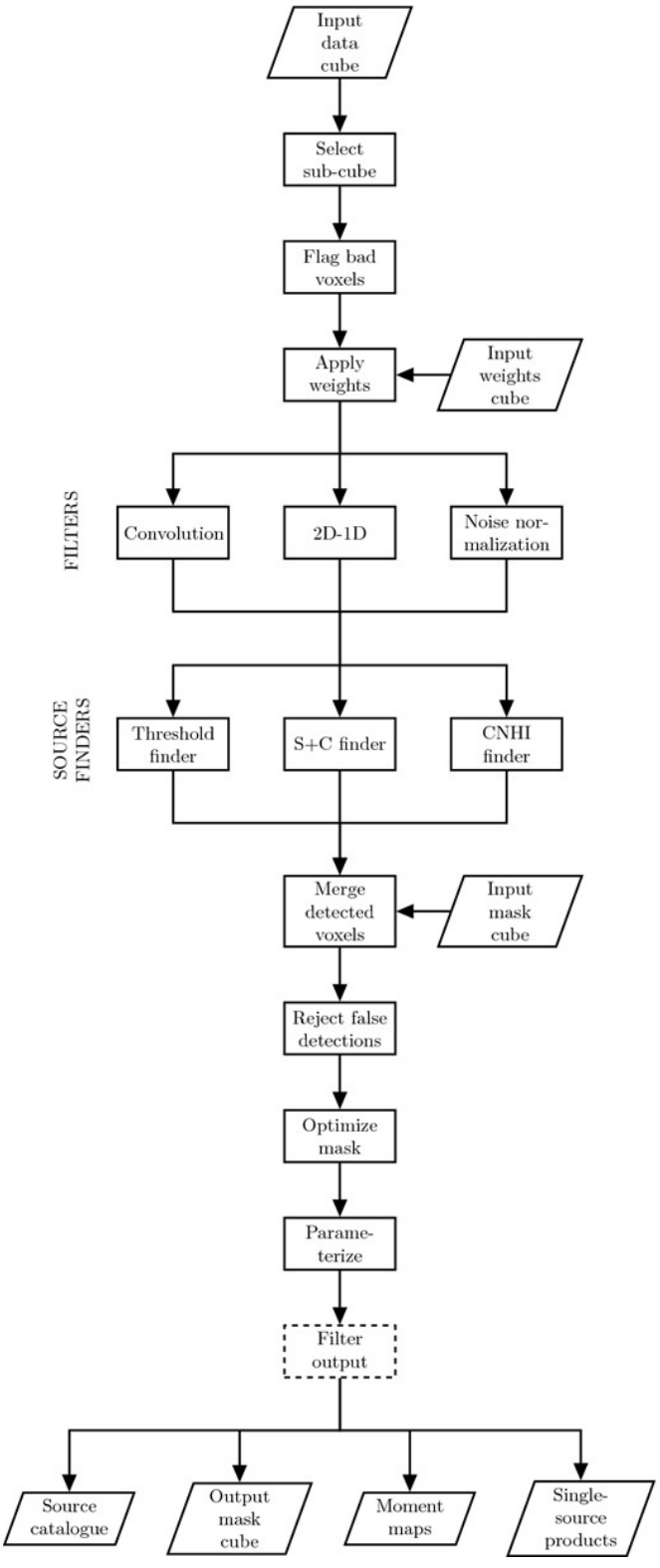


Fig. 1. An illustration of the module pipeline implemented within SoFiA [12].

values, remove noise from an image, highlight objects in an image or classify what is shown in the image. An example of a CNN is shown in Figure 2.

Over the years CNNs have earned their place among the best tools for the job in multiple fields. As shown by the results from the SKA Science data challenge object detection for vast radio astronomy is one of the fields that has been influenced by the power of CNN. Among the top three participating teams in the challenge the top two used a or multiple CNNs in their source finding pipeline. The number one MINERVA used a highly modified YOLO version in union with the Convolutional Hybrid Ad-Hoc pipeline (CHADHOC) and the number two FORSKA-Sweden used 3D U-net [9].

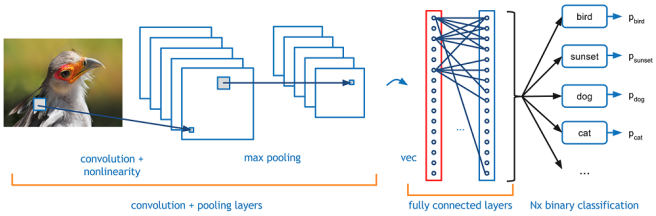


Fig. 2. An example illustration of the architecture of an CNN [5].

2.2.1 YOLO-CIANNA

As stated above MINERVA used a highly modified version of the CNN called YOLO. YOLO is short for you only look once [10], which is the basic principle of this CNN. By reframing object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities the system only has to look once to predict what objects are present and where they are. The CNN is in comparison quite simple. It does not need a complex pipeline because it frames the detection as a regression problem, needs only 1 convolutional network, can simultaneously predict multiple bounding boxes with class probabilities and because of its simplicity is lightning fast. Where other CNNs struggle the YOLO CNN is able to preform detection task in real-time. The simplicity of YOLO is shown in Figure 3. Besides all the good the YOLO CNN also has some limitations. The CNN can only predict 2 bounding boxes and 1 class per each grid cell, has a hard time predicting small objects in groups, struggles in new or unusual aspect ratios or configurations and the loss function treats errors independent of the bounding box size.

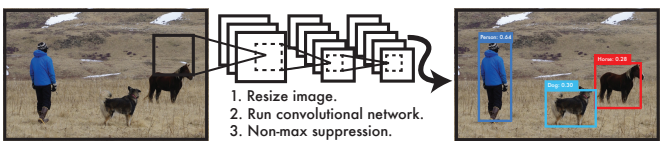


Fig. 3. The YOLO Detection System. From left to right: the image is resized to an image of 448 × 448, a single convolutional network is run with the image and the resulting detections by the model's confidence [10].

The modified YOLO version used by MINERVA is part of the general-purpose CNN framework called CIANNA. This deep learning framework is primarily developed and used for astronomical data analysis [4][9]. The network operates on sub-volumes of 48 × 48 × 192 pixels. Of these sections of 8 × 8 × 16 pixels were mapped to single elements on a 6 x 6 x 12 grid. Per grid element the network could only predict one detection box. Furthermore, the loss function was modified to allow for the prediction of required variables like HI flux for each possible box. The network is made of 21 3D-convolutional layers among which filters that extract morphological properties and filters that force a higher degree feature space while preserving a manageable number of weights to optimize. In some layers a higher stride

value is included to progressively reduce the dimensions to the preferred grid of $6 \times 6 \times 12$. In the last layers error estimation and dropout for regularization are included. The total order of parameters for this network is in the order of 2.3×106. The network was trained on an augmented data set of around 1500 'true' objects with 10 percent set aside for validation. The training set was defined by using the CHADHOC classical detection algorithm among other criteria. As input the network was given a sub-volume containing either at least one true source or a random empty field. Despite the usually fast computational nature of YOLO CNN the modified version used by MINERVA still needed up to 36 hours to train on a single RTX 3090 GPU. The processing of the full 1TB of data also took up to 20 hours.

2.2.2 CHADHOC

This pipeline used by MINERVA has been specifically developed for the SKA Science Data Challenge 2 [9]. Although the pipeline in and of itself is not a CNN two of the three steps make use of CNNs. The first step is a traditional detection algorithm, in the second step true sources among the detections are identified with a CNN and in the last step multiple CNNs are used for source parameter estimation.

In the first step the signal cube is smoothed and a threshold is applied to filter out pixels below a fixed SNR. The remaining pixels are combined with a simple friend-of-friend process. The challenge data is divided into 25 chunks and a catalogue of detections based on the summed source SNR values is produced for each chunk.

In the second step the selection is performed by a CNN. The network is made of 8 layers with inserted between almost every layer Batch normalization, dropouts, and pooling layers. the first five are 3D convolutional layers with 8, 16, 32, 32, and 8 filters. The last three are dense layers 96, 32, and 2 neurons. The network has around 105 parameters. The output of the network is either a true or a false for each detection. The split between true or false is optimised based on an independently set threshold. The training data is an augmented set of the 105 brightest detections in the development cube assigned with a True/False label. A third of the detections were set aside as test data.

For the final step a variation of the CNN used in the previous step is used to predict the source properties. For this CNN augmented versions of the around 1300 brightest sources from the development cube were used to build the learning and test sets.

Since both pipelines accurately detected a slightly different range of sources, for an optimal result the output catalogues of both pipelines was merged.

2.2.3 3D U-Net

The CNN used by FORSKA-Sweden for source finding is 3D U-Net. 3D U-Net is a modified version of the U-Net CNN [15]. In principle U-Net is a CNN designed for medical image segmentation [11], but as proven by FORSKA-Sweden also performs well for object detection for vast radio astronomy. The goal of U-Net is to learn more precise segmentation from very few training images. The segmentation is achieved by keeping track of high resolution features during the standard contracting phase, adding an expanding phase to the network by means of upsampling instead of pooling operators, combining the data of these two steps and finally a convolution layer to convert the new data to a more precise output. See Figure 4 for an illustration of the architecture. To prevent the loss of context information the upsampling phase has a large number of feature channels. There are no fully connected layers in the network and only the valid convolution parts are used. To correctly segment touching objects the network uses a weighted loss which favours background labels between places where cells touch.

The 3D modification of U-Net mainly replaces the input and all the steps for a 3D counterpart: 3D volumes, 3D convolutions, 3D max pooling, and 3D up-convolutional layers [11]. As an extra modification for fast convergence they made use of batch normalization.

FORSKA-Sweden modifications in the encoder used multiple ResNet blocks between each downsampling layer and before each activation a batch normalization layer. Five different resolutions where used for downsampling. In the decoder linear interpolation was used

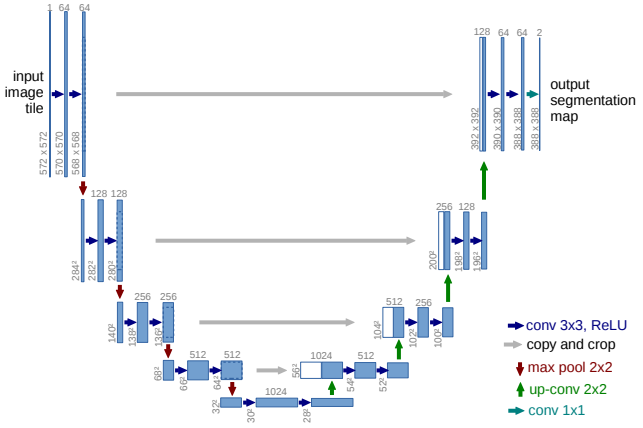


Fig. 4. The architecture behind U-net. The illustration shows an example for the lowest resolution of 32x32 pixels [11].

for upsampling. for each step, excluding the last one were Sigmoid activation was used, the next layers where repeated twice in the mentioned order: convolution, batch normalization and ReLU activation. For training data they used the lower 80 percent of the development cube. From this they sampled batches of 128 cubes of size $32 \times 32 \times 32$ pixels. The other 20 percent was used to validate the results and tune the hyperparameters. Furthermore, a soft dice loss function was used.

2.3 Component trees

A completely different method that might be interesting but was not used in the challenge is component trees. A hierarchical view of connected components in the shape of a tree is what is called a component tree. Such trees can be generated by bounding areas of an image based on some threshold and iterating over them reducing or increasing the threshold, generating a tree. The concept is fairly simple, but very powerful morphological tool that can be used to find objects. In this paper we will look at two version, MTObjects and DISCCOFAN.

2.3.1 MTObjects

MTObjects is a max tree based tool for object detection in astronomical images proposed by Teeninga et al. [13]. An example is shown in Figure 5. The solution was extended by Arnoldus [1]. It implements a varying threshold depending on the properties of the connected components in that set and a filtering method that lowers the threshold for sources that are spread out over a large area. A SExtractor [3], a program that builds a catalogue of objects from an astronomical image, is used to binarize objects at a series of levels. Components on neighbouring levels that only correspond to each other are considered to be the same object. If multiple components in a higher level match to one component in the lower level this component in the lower level should be split. Furthermore the threshold of each level is computed based on the lowest grey level and components below some area threshold are ignored. Arnoldus [1] extension of the tool focuses on adding a smoothing step, improving the classification model and attribute thresholding for large data sets.

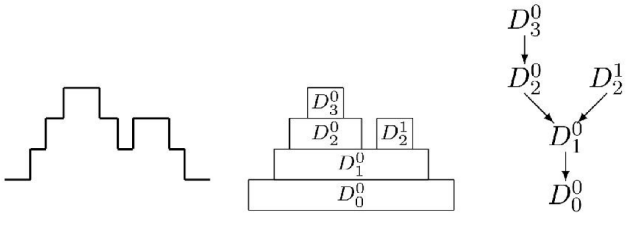


Fig. 5. An example of a 1-D max tree [13].

2.3.2 DISCCOFAN

DISCCOFAN short for Distributed Connected Component Filtering and Analysis [7]. Like MTObjects DISCCOFAN is a component tree based algorithm designed for 2D and 3D Terra-Scale data sets. It is able to perform multiple operators on different scales and it combines shared and distributed memory techniques. DISCCOFAN is a combination and improvement of state of the art algorithms to improve its efficiency and extend its range of data sets. The algorithm is split into four steps, building of the local component trees, optimal selection of border nodes, merging of boundary trees and updating the trees.

3 Framework

While fast algorithms are an important part of handling the fast size of SKA-Scale data the framework on which these algorithms are run is as important. Dodson et al. [6] proposed and performed a test on three platforms to determine how well they would handle and how fast operation within these platforms were on the SKA-Scale data. The paper starts with the four issues of SKA-Scale datasets: cost of the compute environment, transfer of data, data reduction tools used and parallelism. The platforms tested were: a moderately sized cluster (Pleiades), a massive High Performance Computing (HPC) system (Magnus), and the Amazon Web Services (AWS) cloud computing platform. The Pleiades cluster is specifically designed to provide a development platform for r ICRAR's HPC projects. It only contains six-compute nodes, one or two GPUs and has a bandwidth of around 40Gbps. The Magnus platform was created to provide high level scientific computing resources specifically for the Australian research community. The platform has a significantly higher amount of nodes, 1536, of which just 44 were used for this project. For the AWS system the power of the system is flexible depending on the amount of money spent to run the task. For the mentioned project a python script was written that will look for the best price in the specified region. The tests performed, in this order were: system setup, move the data to the system, splitting the files on frequency, inverting the image and finally image concatenation, see Figure 6 for an illustration of the workflow. The Pleiades and Magnus system were similar enough that the same pipeline could be used for both. The AWS cloud platform needed a special pipeline to perform all tests. To capture the result they used two different methods. For the first method they made use of a Linux PROC file system feature. Each process contains a process-specific kernel counter. By periodically looking at this value for the right process one can determine how much progress this process has made in the elapsed time period. Since the first method cannot track every bit of needed information they used the STRACE linux system tool as a second method. The tool gives access to more advanced I/O performance indicators.

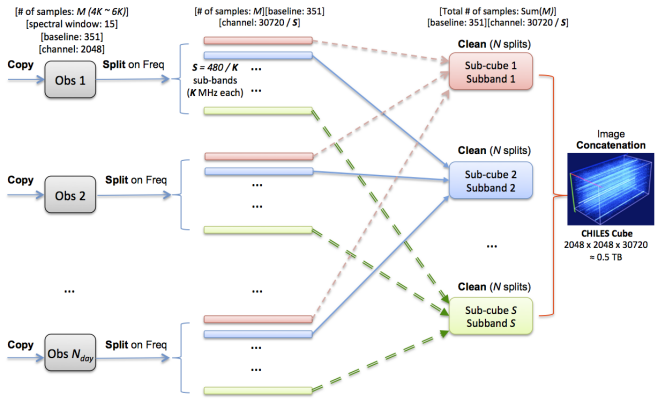


Fig. 6. The workflow used by Dodson et al. [6] to test the three platforms.

4 Results & Discussion

It is clear from the results of the SKA data challenge that both CNN based solutions discussed in this paper performed best with the hybrid implementation created by Team MINERVA being the winner in the challenge [9]. If we take a closer look at the results of the challenge, Figure 7, we notice a huge gap in the scoring between Team Sofia which uses the SoFiA tool and the two teams using a CNN for source detection. While the difference between the first two places is less than a 1000 points the difference between the second and third place is over 5500 points. To make an estimation of where MTObjects would fit in this comparison we need to look at the result mentioned by Barkai et al. [2]. According to their result SoFiA far outperformed MTObjects. Where SoFiA correctly detected 112 real sources MTObjects was only able to detect 26. For the false positives the difference was even greater with SoFiA only having 427 and MTObjects hitting a staggering 3488. With these results it is safe to say that MTObjects comes nowhere near the performance of the first and second place contenders of the SKA data challenge. With MTObjects scoring this low the expectation is that disccofan being a component tree based approach will rank similarly. It might perform better than expected but, it is difficult to place it anywhere on the scoring board.

Team name	Score	N_d	N_m	R	C	A
MINERVA	23254	32652	30841	0.945	0.132	0.81
FORSKA-Sweden	22489	33294	31507	0.946	0.135	0.77
Team SOFIA	16822	24923	23486	0.942	0.101	0.78

Fig. 7. A subset of the scores from the SKA data challenge [9].

For the framework to run the solutions on it depends on which factors are important. Looking at the results mentioned in Dodson et al. [6] AWS has the lowest data transfer of 1Gb but has the highest bandwidth ~300MB/s, IOPS ~1000 and fastest completion time of 96h. If the focus is purely on using the system once or twice AWS is the better solution because the operation cost is comparatively cheap with only $2000 and no capital cost. If we look at the use case where the system would be used regularly AWS is still the better option since the operation cost of the Magnus system is more than 1.5 times as expensive as the operation cost of AWS. Although the Pleiades system does not have an operational cost the capital cost of the system is 25 times more than the AWS operation cost while the processing time is more than 10 times as long as on the AWS system. One can argue that the extra operation cost the AWS system eventually will cost is negligible in comparison with the gain in processing power.

5 Conclusion

With the data currently available it is hard to give a concise conclusion and more research is definitely needed. However, for now the implementation created by team MINERVA is the best performing solution discussed in the paper. Their implementation outperformed the CNN used by team FORSKA-Sweden, SoFiA and in comparison both component trees based solutions. For the framework the AWS system would be the best choice. In the long term it would have a higher operation cost than the Pleiades system however, this is made insignificant by the huge improvement in processing time.

6 Future Work

Going forward DISCCOFAN could be properly measured against one of the mentioned methods created for the SKA data challenge to get an accurate measurement of where it lies performance wise. Furthermore, a test of some of these solutions on AWS and potentially other systems would neatly show how ready the field is for handling SKA scale data.

7 Acknowledgments

The author of this paper wishes to thank the anonymous reviewers and the expert reviewer M.H.F. Wilkinson for their helpful feedback. They

also wish to thank the teacher of the course, R. Smedinga, for their support. Lastly, the author wants to thank their significant other for helping them through the stressful periods while writing this paper.

REFERENCES

[1] C. Arnoldus. A max-tree-based astronomical source finder. *Master's thesis, University of Groningen*, 2015.

[2] U. Barkai, M. Verheijen, E. Talavera Martínez, and M. Wilkinson. A comparative study of source-finding techniques in hi emission line cubes using sofia, mtobjects, and supervised deep learning. *Astronomy & astrophysics*, 670, Feb. 2023.

[3] E. Bertin and S. Arnouts. SExtractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 117:393–404, June 1996.

[4] D. Cornu. Deyht/cianna: Convolutional interactive artificial neural networks by/for astrophysicists, Aug 2019.

[5] A. Deshpande. A beginner's guide to understanding convolutional neural networks, Jul 2016.

[6] R. Dodson, K. Vinsen, C. Wu, A. Popping, M. Meyer, A. Wicenec, P. Quinn, J. van Gorkom, and E. Momjian. Imaging ska-scale data in three different computing environments, 2015.

[7] S. Gazagnes and M. Wilkinson. Distributed connected component filtering and analysis in 2-d and 3-d tera-scale data sets. *Ieee transactions on image processing*, 30:3664–3675, 2021.

[8] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[9] P. Hartley, A. Bonaldi, R. Braun, J. N. H. S. Aditya, S. Aicardi, L. Alegre, A. Chakraborty, X. Chen, S. Choudhuri, A. O. Clarke, J. Coles, J. S. Collinson, D. Cornu, L. Darriba, M. D. Veneri, J. Forbrich, B. Fraga, A. Galan, J. Garrido, F. Gubanov, H. Håkansson, M. J. Hardcastle, C. Heneka, D. Herranz, K. M. Hess, M. Jagannath, S. Jaiswal, R. J. Jurek, D. Korber, S. Kitaeff, D. Kleiner, B. Lao, X. Lu, A. Mazumder, J. Moldón, R. Mondal, S. Ni, M. Önnheim, M. Parra, N. Patra, A. Peel, P. Salomé, S. Sánchez-Expósito, M. Sargent, B. Semelin, P. Serra, A. K. Shaw, A. X. Shen, A. Sjöberg, L. Smith, A. Soroka, V. Stolyarov, E. Tolley, M. C. Toribio, J. M. van der Hulst, A. V. Sadr, L. Verdes-Montenegro, T. Westmeier, K. Yu, L. Yu, L. Zhang, X. Zhang, Y. Zhang, A. Alberdi, M. Ashdown, C. R. Bom, M. Brüggen, J. Cannon, R. Chen, F. Combes, J. Conway, F. Courbin, J. Ding, G. Fourestey, J. Freundlich, L. Gao, C. Gheller, Q. Guo, E. Gustavsson, M. Jirstrand, M. G. Jones, G. Józsa, P. Kamphuis, J.-P. Kneib, M. Lindqvist, B. Liu, Y. Liu, Y. Mao, A. Marchal, I. Márquez, A. Meshcheryakov, M. Olberg, N. Oozeer, M. Pandey-Pommier, W. Pei, B. Peng, J. Sabater, A. Sorgho, J. L. Starck, C. Tasse, A. Wang, Y. Wang, H. Xi, X. Yang, H. Zhang, J. Zhang, M. Zhao, and S. Zuo. SKA Science Data Challenge 2: analysis and results. *Monthly Notices of the Royal Astronomical Society*, 523(2):1967–1993, 05 2023.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016.

[11] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[12] P. Serra, T. Westmeier, N. Giese, R. Jurek, L. Flöer, A. Popping, B. Winkel, T. van der Hulst, M. Meyer, B. S. Koribalski, L. Staveley-Smith, and H. Courtois. SoFiA: a flexible source finder for 3D spectral line data. *Monthly Notices of the Royal Astronomical Society*, 448(2):1922–1929, 02 2015.

[13] P. Teeninga, U. Moschini, S. Trager, and M. Wilkinson. Bi-variate statistical attribute filtering: a tool for robust detection of faint objects. 11 2013.

[14] T. Westmeier, S. Kitaeff, D. Pallot, P. Serra, J. M. van der Hulst, R. J. Jurek, A. Elagali, B.-Q. For, D. Kleiner, B. S. Koribalski, K. Lee-Waddell, J. R. Mould, T. N. Reynolds, J. Rhee, and L. Staveley-Smith. SoFiA 2 – an automated, parallel HI source finding pipeline for the WALLABY survey. *Monthly Notices of the Royal Astronomical Society*, 506(3):3962–3976, 07 2021.

[15] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016.

Integrating frequency information in few-shot learning: A Comparative Overview

Joy Kwant, Tomas de Vries

Abstract— Deep Neural Networks have had many achievements in computer vision tasks. However, Deep Neural Networks mostly rely on large-scale datasets to achieve good results. Few-shot classification aims to develop a discriminative feature representation capable of recognizing unseen classes using a limited number of labeled samples. Previous studies have focused on using spatial images or videos as input. However recent studies have shown that using frequency information can improve the results of few-shot classification.
In this paper the focus is on comparing different existing methods proposed in recent studies on integrating frequency information, with the goal of augmenting classification accuracy. This includes methods using a novel Frequency-Guided Few-shot Learning framework (FGFL), Discrete Cosine Transformations and structures of well-known neural networks, such as ResNet-50. We will discuss and judge these methods based on how much the respective methods improve results compared to the spatial approach.
When looking at the current state of the art it becomes clear that integrating frequency information can significantly improve the accuracy of image classification using few-shot learning for varying sizes of training data compared to integrating spatial information.

Index Terms—Image classification, few-shot learning, frequency information, frequency-guided few-shot learning framework, discrete cosine transformation.

1 Introduction

Deep learning has been making a lot of progress in image processing and computer vision over the last couple of years [12, 6, 16]. Deep learning is a subset of machine learning. Where machine learning is focused on running algorithms that parse data and learn from itself. Deep learning is inspired by the human brain and learns to recognize complex patterns by using artificial neural networks with many layers [9]. The high accuracy that deep learning can achieve causes it to be the main focus for image processing and computer vision.

However, it also has drawbacks. To achieve desirable results, it is necessary to train the algorithm with a very large amount of data. Unfortunately, there are scenarios where there is simply not enough data to train the model on. For example, when classifying rare diseases [3].

Another drawback is the constraint of computing resources and memory limitations. This causes most deep learning models to use low resolution RGB-images as input (e.g. 244x244) [24]. While most modern cameras have a much higher resolution.
The direction of Computer Vision that focuses on both problems described above is few-shot learning.

The goal of few-shot learning is to learn a discriminative feature representation to classify novel data [4]. The K-way N-shot few-shot learning is able to mimic large-scale data tasks by training a low number of labeled examples (N-shot) per class (K-way), requiring less data and less dependency on capacity and computational limit.

Before, various few-shot classification methods used the image spatial domain to classify the novel data [11, 26]. Nevertheless, the methods were not able 'to achieve stable performance' [4]. In Computer Vision, besides the image spatial feature, the frequency domain has become essential for several tasks, e.g. image classification [15], deep fake recognition [5] and domain generalization [8]. Research shows that frequency information can also be essential for few-shot learning [3].

This paper focuses on analyzing different methods that use frequency information for few-shot learning proposed in [3], [4], and [24].

- *Joy Kwant is with University of Groningen, E-mail: j.kwant@student.rug.nl.*
- *Tomas de Vries is with University of Groningen, E-mail: t.de.vries35@student.rug.nl.*

This paper will look into the following research questions:

- How can we use frequency information to improve image classification?
- What are different methods to use frequency information to improve few-shot learning?

We hypothesize that the research questions should give us the following:

- A way to integrate frequency information in image classification that improves image classification using spatial information.
- An overview of different methods that integrate frequency information in few-shot image classification from which we can conclude an optimal method.

In section 2 we will look into the current state of the art that looks into the extraction of frequency information from RGB datasets to improve classification tasks. In section 3 we will compare the different methods for few-shot classification and propose an improvement. In section 4 we will give our conclusion.

2 State of the art

In this section, we will talk about existing research that has focused on image classification using few-shot learning. The main focus is on frequency information in few-shot learning, which is discussed in sections 2.3 and 2.4.

We will discuss three main papers, before we go into detail about their methods we will look at other researches that have looked into integrating frequency information in image classification with large-scale datasets.

The first paper 'Learning in the Frequency Domain' [24] proposed a pipeline to convert RGB images to the frequency domain by using the Discrete Cosine Transform and selecting the most important frequency channels.

After that, we look into the paper that builds upon the proposed method of the first paper: 'Few-Shot Learning by Integrating Spatial and Frequency Representation' [3]. By combining spatial information with an altered version of the pipeline to get frequency information they look into few-shot classification.

Lastly, we will discuss the paper 'Frequency Guidance Matters in Few-Shot Learning' [4] that proposes a novel framework that uses masks, extracted from the frequency information, for the few-shot classification.

2.1 Frequency information in image classification

Integrating frequency information for image classification in Computer Vision has been proposed in several research papers [18, 27, 19, 10]. In this section, we will look at some researches that show the potential of the integration of frequency information in image classification.

The research in 'Global filter networks for image classification' [15] proposes a standard transformer architecture but with a Multi Layer Perceptron-like model to prevent a heavy self-attention layer. It proposes to use a Fourier transform with learnable filters to globally interchange information among the data points in the Fourier domain. This gave accuracy up to 96.2% with the ImageNet dataset while still being very efficient with log-linear complexity.

'High-frequency component helps explain the generalization of convolutional neural networks' [23] researches how frequency information affects the generalization behavior of Convolutional Neural Networks. They came to the conclusion that CNN may capture high frequency components that are not aligned with the human visual preference. However, they found that capturing these components sometimes corresponds to improved accuracy.

Both papers show significant improvements compared to image classification using spatial information. In the next section, we will go more in-depth on a method to integrate frequency information for image classification. We look into this in more detail because it is altered in section 2.3 to be integrated with few-shot learning.

2.2 Learning in the frequency domain

The paper discussed in this section, 'Learning in the Frequency Domain' [24], proposes a generic method for learning in the frequency domain.

2.2.1 Methods

Preprocessing RGB images Conventionally, it is common for RGB images to be preprocessed on a CPU before being transmitted to a GPU/AI accelerator for real-time interactivity. Uncompressed RGB images require a high communication bandwidth, which can be a bottleneck to an application. Therefore RGB images are often downsampled, likely causing information loss.

The paper proposes to preprocess RGB images like in figure 1. This pipeline shows a process where information in the frequency is used instead of information in the spatial dimension. To effectively use the information from the frequency domain the first step is to convert the input RGB images to the YCbCr color space, where Y is luminance, Cb and Cr are colors.

Discrete Cosine Transform To get the frequency information the spatial information is converted using the Discrete Cosine Transform (DCT). DCT is a transformation that helps separate the image into parts of different importance represented in the frequency domain [1]. It is known to be more efficient than Fast Fourier Transform with fewer data points, which is preferable in few-shot learning. Also, research has shown that the results using DCT are close to the results of Karhunen—Loève Transform (KLT), which are considered to be optimal [1].

The two-dimensional DCT information is partitioned in patches of size 8x8 [24]. This patch size was chosen because it is the standard of JPEG compression. The reason why this is the standard for JPEG compression mainly lies in issues like memory requirements, number of operations per sample, and amenability to hardware or software implementation [13]. Components of the same frequency are grouped into one channel to create a cubic representation as seen in the DCT reshape step in figure 1. Therefore every component, Y, Cb, and Cr, gives 8x8=64 channels. Resulting in 192 channels ($C = 192$) in total for the frequency domain. If the input of an image is $H \times W \times C$, the output will be $H/8 \times W/8 \times C * 64$, resulting in the same input size.

Frequency channels Different channels of the input features are at different channels of the frequency information, leading to the conclusion that some frequency channels can be less informative for tasks like image classification, object detection, and instance segmentation.

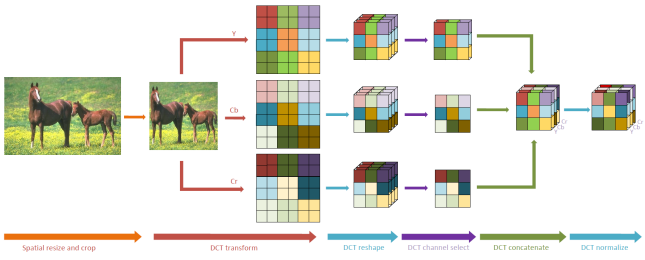


Fig. 1. Pre processing pipeline [24].

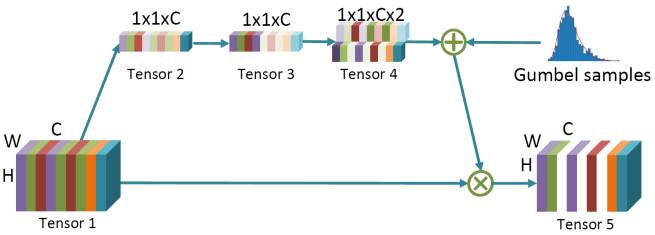


Fig. 2. Dynamic gate module that generates the binary decisions. The white color channels of Tensor 5 indicate unselected channels [24].

Since these features are not important, removing them shall not result in a worse performance. However, it will lead to less computational complexity and lower communication bandwidth.

To optimally remove redundant/non-informative frequencies the paper proposes a learning-based frequency selection mechanism. They use a dynamic gate module that assigns a binary score to each frequency channel. Where unimportant channels should have a value of 0 and important channels should have a value of 1. This dynamic gate is shown in figure 2. Tensor 1 transforms into Tensor 2 by an average pooling layer. Tensor 2 is transformed into Tensor 3 by a convolution layer. These transformations use the information from the channels to suppress trivial features and emphasize informative features. Tensor 3 is transformed into Tensor 4 by multiplying every element in Tensor 3 with two trainable parameters. Tensor 4 has a size of $1 \times 1 \times C \times 2$, so it has a fourth dimension. The values in this dimension are normalized and function as the probability for the channel to be 0 or 1. To obtain Tensor 5 the input frequency is point-wise multiplied with either 0 or 1 depending on the probabilities in the fourth dimension of Tensor 4. Tensor 5 is the final tensor that consists of only the important features.

2.2.2 Results

The paper looks into the importance of different frequencies to understand the pattern of frequency channel activation. The results obtained from the learning-based feature selection give an estimation of the importance of each frequency channel. To understand the results, they were visualized as a heat map in figure 3. The numbers in the boxes correlate to their frequency index, where lower frequencies have a lower frequency index. The heat map value correlates to the probability a frequency channel will be selected for inference across all validation images.

Figure 3 displays a few patterns. The first one is that low-frequency channels are more likely to be selected than high-frequency channels, indicating that low-frequency channels are more informative. Secondly, frequency channels from the Y component are more likely to be selected than channels from the Cb and Cr components. Meaning that luminance is more informative than color for image classification. The ResNet-50 model is trained with 192 frequency channel inputs on the image classification task using the approach described above. The dynamic gate module responsible for channel selection is trained along with the ResNet-50 model. It was concluded earlier that the top left frequency channels were most important. Because of this, the pa-

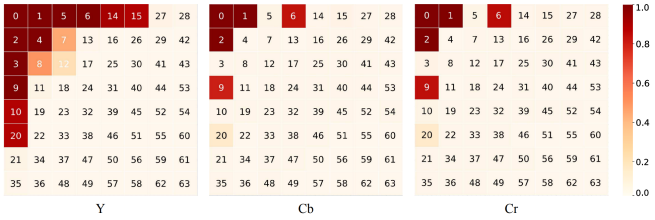


Fig. 3. Heat map of frequency importance for Y, Cb, and Cr for the ImageNet dataset [24].

ResNet-50	#Channels	Size Per Channel	Top-1	Top-5	Normalized Input Size
RGB	3	224×224	75.780	92.650	1.0
YCbCr	3	224×224	75.234	92.544	1.0
DCT-192 [17]	192	28×28	76.060	93.020	1.0
DCT-192 (ours)	192	56×56	77.194	93.454	4.0
DCT-24D (ours)	24	56×56	77.166	93.560	0.5
DCT-24S (ours)	24	56×56	77.196	93.504	0.5
DCT-24T (ours)	24	56×56	77.148	93.326	0.5
DCT-48S (ours)	48	56×56	77.384	93.554	1.0
DCT-48T (ours)	48	56×56	77.338	93.614	1.0
DCT-64S (ours)	64	56×56	77.232	93.624	1.3
DCT-64T (ours)	64	56×56	77.280	93.456	1.3

Table 1. Results for different frequency channel selections [24].

per looks into different frequency channel selections. The results with different frequency channel selections are shown in figure 1.

Where every "D" method uses the dynamic learning-based channel selection, every "S" method is the top left square, and every "T" method is the top left triangle, all with the respective size that is indicated. For DCT-24, 14, 5, and 5 channels were selected for Y, Cb, and Cr respectively. The differences between DCT-24D, DCT-24S, and DCT-24T are minimal. This confirms that the top-left frequencies of the heat map are the most important. More importantly, this paper also shows that the use of frequency information can improve the system's accuracy. And figure 1 even shows that a lower selection of frequency channels can improve results.

2.3 Integrating Spatial and Frequency Representation

The paper described in section 2.2 inspired future research to look into implementing frequency information for image processing. The paper 'Few-Shot Learning by Integrating Spatial and Frequency Representation' [3] builds upon this and will be discussed in this section. For a K-way N-shot learning problem, with K novel classes and N labeled support samples, the focus is to classify a query sample into one of these K support classes. The results will be measured using the accuracy of this classification.

2.3.1 Methods

Integrating spatial and frequency representation The paper built upon the DCT pipeline given in the paper discussed in section 2.2. This pipeline can be seen in figure 4.

As shown in the figure, the pipeline consists of five steps. Firstly, as seen in step (a) the image is preprocessed by cropping, rotating and translating to get images of size S_{image} (e.g. 448x448). In step (b) the RGB image is converted to a YCbCr image. YCbCr, consists of the luminance value Y and the color values Cb and Cr. The conversion is done using a 4:2:0 color sub-sampling, which halves the horizontal and vertical size of the color values. Step (c) performs the Discrete Cosine Transform. This is done after the channels are divided in patches of $S_{dct} \times S_{dct}$ where $S_{dct} < S_{image}$ (e.g. 8x8). This results in patches of $S_{dct} \times S_{dct}$ with frequency information.

By grouping the same frequency to one sub-channel the patches can be converted to cubes as seen in step (d). To get from (d) to (e), they first get more impactful frequency sub-channels, as obtained in [24]. These will then be concatenated into one frequency cube. To get Cb

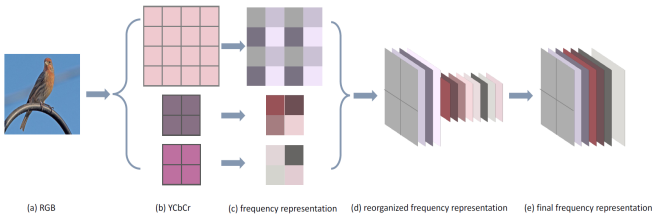


Fig. 4. (a) images after preprocessing, e.g. 448 × 448 × 3. (b) transform RGB to YCbCr images, Y: 448 × 448, Cr and Cb: 224 × 224. (c) frequency representation after DCT transformation with a filter, e.g. 8x8, (d) reorganize frequency representations by frequency channels, Y: 56×56×64, Cb and Cr: 28×28×64. (e) upsample all Crdct and Cbdct frequency channels to the same size with Ydct channels and keep those frequency selected channels. Y: 56 × 56 × 16, Cb and Cr: 28 × 28 × 4 [3].

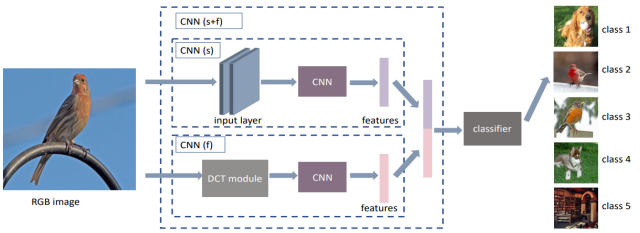


Fig. 5. The structure of the proposed CNN framework [3].

and Cr to the size of Y, they will be upsampled with interpolation. This gives the final frequency representation.

This frequency representation is then used as input for a CNN. The framework of the network is shown in figure 5. The upper network, "CNN (s)", where (s) means the spatial domain, denotes a regular image classification network where the input layers are drawn before the backbone from the whole CNN. In the lower network, "CNN (f)", where (f) means frequency domain, the images will go through the DCT module first to generate the frequency representation before being supplied to the CNN backbone. Finally, both features are concatenated to generate the final classification score, which is the whole output network, "CNN (s+f)".

Datasets To get the results, the paper uses the *mini*ImageNet [21], CUB [22] and CIFAR-FS datasets [2]. These are popular datasets used for few-shot learning. Of these sets *mini*ImageNet contains 100 classes, 64 are used for training, 16 for validation, and 20 for testing. CUB has 200 classes, 150 for training, 20 for validation and 50 for testing. and CIFAR-FS has 100 classes, 64 for training, 16 for validation and 20 for testing. All images are of size 32x32.

2.3.2 Results

The results of the paper will give an estimation of the generalization to the existing learning frameworks.

To analyze the performance of integrating frequency information to few-shot learning the paper uses the information resulting from the DCT-pipeline as input for other few-shot learning frameworks. The frameworks that were used were: the Matching Network (MN) [20] (metric-based algorithm), S2M2R [14] (pre-train and finetune), and PT+MAP [7] (post-process S2M2R features). Figure 2 shows the results. The results show that implementing information from both the spatial domain and the frequency domain improves the results for every method.

PT+MAP [7] uses the learned features and applies them to the network S2M2$_R$ [14]. Which is designed in a preprocessing way, making it compatible for all networks. And it results in a high accuracy com-

backbone	method	accuracy on *mini*ImageNet 1-shot	5-shot
ResNet10	MN [12]#	54.49±0.81	68.82±0.65
	MN (s)*	52.98±0.21	72.41±0.16
	MN (f)	55.98±0.20	74.17±0.16
	MN (s+f)	**57.32±0.21**	**76.27±0.16**
		+4.34	*+3.86*
WRN-28-10	S2M2$_R$[25]	64.93±0.18	83.18±0.11
	S2M2$_R$ (s)*	63.09±0.17	80.88±0.11
	S2M2$_R$ (f)	63.03±0.18	80.80±0.11
	S2M2$_R$(s+f)	**66.88±0.18**	**84.26±0.10**
		+3.79	*+3.38*
WRN-28-10	PT+MAP [41]	82.92±0.26	88.82±0.13
	PT+MAP (s)*	80.73±0.24	87.81±0.13
	PT+MAP (f)	82.04±0.23	88.68±0.12
	PT+MAP (s+f)	**84.81±0.22**	**90.62±0.11**
		+4.08	*+2.81*

Table 2. Results after integrating features from both the spatial and frequency domains to existing methods on *mini*ImageNet. The highest accuracy is highlighted [3].

method	backbone	***mini*ImageNet** 1-shot	5-shot	**CUB-200-2011** 1-shot	5-shot
ProtoNet* [19]	ConvNet	50.37±0.83	67.33±0.67	66.36±1.00	82.03±0.59
MAML* [13]	ConvNet	50.96±0.92	66.09±0.71	66.26±1.05	78.82±0.70
RelationNet* [15]	ConvNet	51.84±0.88	64.55±0.70	64.38±0.94	80.16±0.64
S2M2$_R$ [25]	WRN-28-10	64.93±0.18	83.18±0.11	80.68±0.81	90.85±0.44
AFHN [42]	ResNet18	62.38±0.72	78.16±0.56	70.53±1.01	83.95±0.63
DPGN [43]	ResNet12	67.77±0.32	84.60±0.43	75.71±0.47	91.48±0.33
DeepEMD-sampling [44]	ResNet12	68.77±0.29	84.13±0.53	79.27±0.29	89.80±0.51
PT+MAP [41]	WRN-28-10	82.92±0.26	88.82±0.13	91.55±0.19	93.99±0.10
PT+MAP (s+f) (ours)	WRN-28-10	**84.81±0.22**	**90.62±0.11**	**95.48±0.13**	**96.70±0.07**

Table 3. Results of the paper's method along with other state-of-the-art results on *mini*ImageNet, CUB, and CIFAR-FS. The highest accuracy is highlighted [3].

pared to other state-of-the-art frameworks. So to compare the results to other state-of-the-art methods the paper implements their method with PT+MAP. When implementing PT+MAP with the spatial domain and frequency domain the paper denotes it as 'PT+MAP (s+f)'. The results can be seen in figure 3.

When comparing the results of PT+MAP (s+f) with other methods it is clear that it can increase the accuracy of the state-of-the-art methods. This holds for all datasets that were tested. For *mini*Imagenet, PT+MAP (s+f) increases the best accuracy by 1.89% and 1.8% for 5-way 1-shot and 5-way 5-shot, respectively. For the CUB dataset, the accuracy is increased by 3.93% and 2.71% respectively. An important thing to note is that the image size is small for the CIFAR-FS dataset, with a size of only 32x32. This could prevent a good increase in accuracy. However, we can still observe an increase of 1.81% and 1.48% for the two tasks, respectively.

2.4 Frequency-Guided Few-shot Learning

The paper 'Frequency Guidance Matters in Few-Shot Learning' [4] developed a novel Frequency-Guided Few-shot Learning framework (named FGFL) for the few-shot classification in the spatial domain, using the Discrete Cosine Transformation (DCT) for transforming to the frequency domain.

Their framework consists of two branches for feature extraction; the frequency and spatial domain, and a multi-level metric for obtaining frequency components used for classification. The frequency domain performs the creation of frequency masks which are used 'to guide the training in the spatial domain'. The spatial domain is used for the training and testing in the network. Figure 6 shows an overview of this framework.

2.4.1 Methods

Frequency domain In the frequency domain branch, the framework obtains a 'task-specific attention mask' for each few-shot task, generated with the use of frequency components.

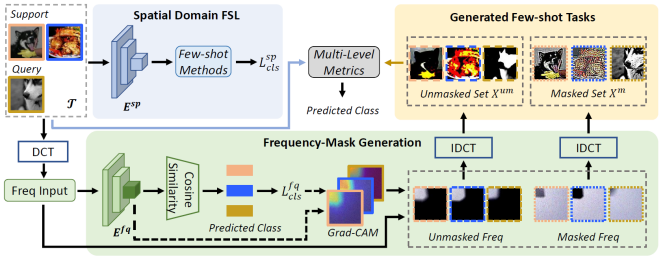


Fig. 6. Overview of the proposed Frequency-Guided Few-shot Learning framework (FGFL) [4].

The frequency domain branch will first use the DCT to acquire the frequency domain D of the input X and producing the corresponding attention map by Grad-CAM [17]. With the cosine similarity between each class prototype and sample for each given K-way N-shot task, the classification score is computed. The computed gradients of the classification score with respect to the feature map activation, are passed through a global average pooling layer to acquire the importance of the activation map, which leads to the prediction of class c of the input. Then a heatmap A_c is created with the ReLU activation function, contemplating the positive affected frequency components.

Lastly, the frequency mask is created with the heatmap A_c and the Sigmoid activation function. The mask will be applied to the input of the frequency domain D and converted back to the spatial domain, generating the unmasked spatial image X^{um} and a masked spatial image X^m. These spatial images are used to 'guide the few-shot learning training in the special domain'.

Multi-level metrics The multi-level metrics of the FGFL captures 'the class discriminative information' in the spatial network with the created two-ranking losses: 'sample-wise triplet loss' (L_{sw_tri}), using the original X_i, unmasked X_i^{um} and masked images X_i^m and the 'class-wise contrastive loss'(L_{cw_ctr}), using support and query sets of the original images (X_i, S and Q) and masked images (X_i^m, S_m, Q_m). For the computation of the augmented loss L_{aug}, the unmasked images were used. The illustration of the multi-level metrics can be seen in figure 7.

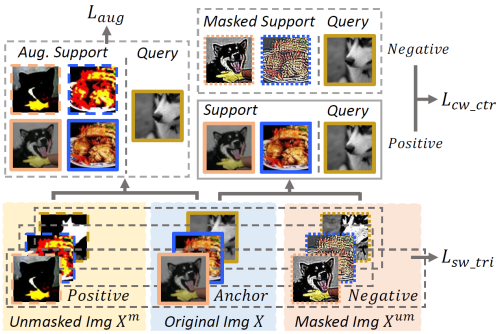


Fig. 7. Illustration of the proposed multi-level metrics with the unmasked images X^m, the masked image X^{um} and the orginal image X, and the losses class-wise contrastive L_{cw_ctr}, the sample-wise triplet loss L_{sw_tri} and augmented loss L_{aug} [4].

Few-shot classification The total few-shot classification function is as follows:

$$L_{total} = L_{cls}^{sp} + L_{cls}^{fq} + \lambda_1 \cdot L_{sw\_tri} + \lambda_2 \cdot L_{cw\_ctr} + \lambda_3 \cdot L_{aug} \quad (1)$$

In the formula, L_{cls}^{sp} and L_{cls}^{fq} correspond to the cross-entropy loss for few-shot classification in the spatial and frequency branch, respec-

tively and λ_1, λ_2, and λ_3 are the weighting parameters of the corresponding losses, respectively.

Set-up They performed their framework on the datasets that are relevant in our study: *mini*ImageNet [21] and CUB [22].

They combined their proposed framework FGFL with FEAT [25] and adopted the ResNet-12 network as a 'backbone' in each branch. They experimented with query sets 5-way 1-shot and 5-way 5-shot, constructed from fifteen randomly selected samples of each class and the image size of 84x84.

2.4.2 Results

They performed experiments on the standard few-shot settings 'general classification', shown in figure 4 and the 'fine-grained classification', shown in figure 5.

The results show that the FGFL 'can improve the performance of few-shot methods.' The FGFL shows the highest accuracy percentage for the 5-shot setting with the dataset *mini*ImageNet and second best for the 1 shot setting. The results with the CUB dataset show that the FGFL is second best in both shot settings and improves the accuracy of the few-shot method FEAT.

Method	***mini*ImageNet**	
	5-way 1-shot	5-way 5-shot
ProtoNet [36] †	61.83 ± 0.20	79.86 ± 0.14
Meta-Baseline [5]	63.17 ± 0.23	79.26 ± 0.17
Good-Embed [38]	64.82 ± 0.60	82.14 ± 0.43
DeepEMD [58]	65.91 ± 0.82	82.41 ± 0.56
FRN [46]	66.45 ± 0.19	82.83 ± 0.13
FEAT [55] †	66.52 ± 0.20	81.46 ± 0.14
BML [60]	67.04 ± 0.63	83.63 ± 0.29
IEPT [59]	67.05 ± 0.44	82.90 ± 0.30
MELR [9]	67.40 ± 0.43	83.40 ± 0.28
MCL-Katz [25]	67.51	83.99
CSEI [21]	67.59 ± 0.83	81.93 ± 0.36
Meta DeepBDC [48]	67.83 ± 0.43	84.46 ± 0.28
DFR [6]	68.12 ± 0.81	82.79 ± 0.56
Yang et al. [54]	**70.19 ± 0.46**	**84.66 ± 0.29**
Ours	**69.14 ± 0.80**	**86.01 ± 0.62**

Table 4. Comparison of methods for general few-shot image classification, using the Average classification accuracy (%) on *mini*ImageNet datasets with the ResNet-12 as backbone. The best and second best results under each setting and dataset are highlighted as Red and Blue, respectively [4].

Method	Backbone	**CUB**	
		5-way 1-shot	5-way 5-shot
MELR [9]	ConvNet-4	70.26 ± 0.50	85.01 ± 0.32
ProtoNet [36] †	ResNet-12	72.25 ± 0.21	87.47 ± 0.13
DeepEMD [58]	ResNet-12	75.65 ± 0.83	88.69 ± 0.50
FEAT [55] †	ResNet-12	75.68 ± 0.20	87.91 ± 0.13
BML [60]	ResNet-12	76.21 ± 0.63	90.45 ± 0.36
Good-Embed [38] ‡	ResNet-18	77.92 ± 0.46	89.94 ± 0.26
DFR [6]	ResNet-12	78.07 ± 0.79	89.74 ± 0.51
VFD [49]	ResNet-12	79.12 ± 0.83	91.48 ± 0.39
FRN [46] †	ResNet-12	**82.90 ± 0.19**	**92.61 ± 0.10**
Ours	ResNet-12	**80.77 ± 0.90**	**92.01 ± 0.71**

Table 5. Comparison of methods for fine-grained few-shot image classification, using the Average classification accuracy (%) on CUB datasets. The best and second best results under each setting and dataset are highlighted as Red and Blue, respectively [4].

3 Discussion

Combining the results we can see that there are several ways to improve image classification using frequency information. The paper discussed in section 2.2 showed that frequency information can be beneficial compared to just using RGB or YCbCr information. However, it does not go into detail on the main focus of this paper: Integrating frequency information in few-shot learning. The results were obtained by using a big dataset, ImageNet, and therefore are not representable for few-shot learning. However, it is worth noting that this paper, along with others that integrate frequency information [23, 15, 18, 27, 19, 10], inspired researchers to look into integrating frequency information in few-shot learning.

The paper discussed in section 2.3 built further on this idea. They implemented an altered version of the pipeline from paper [24] to test it with few-shot learning. When comparing this to the other paper, it should be kept in mind that different datasets were used and therefore there can not be a direct comparison. However, it is still noteworthy to mention that the paper discussed in section 2.3 managed to get an accuracy of 96.7% for the CUB-200-2011 dataset and the paper from section 2.2 got an accuracy of 93.454% with ImageNet. This improvement might be due to the different datasets, but it is worth noting that both methods have a very high accuracy.

The third paper in section 2.4 obtains frequency components from frequency masks for the few-shot classification. Their proposed framework performed better than current few-shot methods, and even has the ability to improve few-shot method FEAT. However, the FGFL is not the best method, when using the different shot-settings.

Table 3 shows that they [3] can achieve an optimal accuracy of 80.77% for *mini*ImageNet and 96.70% for CUB with the 5-shot setting, while table 4 and 5 show optimal accuracies of only 86.01% and 92.01% for the other method [4] and the respective datasets using the 5-shot setting. The better results of paper [3] are likely caused by the use of convolutional neural networks for frequency information and spatial information individually and combining these.

For future work, since the Frequency-Guided Few-shot Learning framework [4] (FGFL) provides significant results in the field it would be interesting to look into the results of combining the FGFL with predictions using spatial information and a convolutional neural network. This combines the most important elements of the two papers [3, 4] and might result in significant improvements.

4 Conclusion

We analyzed multiple papers that integrate information from the frequency domain for image classification.

When looking at the papers it becomes clear that they all manage to improve image classification when compared to classification using information from the spatial domain. These improvements led to research papers that use frequency information for few-shot image classification.

When comparing the different methods we see varying results but they all show that frequency information provides an improvement compared to the same methods when only using spatial information. In section 1 we discussed the need for large datasets for deep learning.

The promising results of the state of the art show that integrating frequency information in few-shot learning has the potential to reduce the need for large datasets in deep learning. Future research might be able to find even better ways to extract more informative frequency information or better ways to apply this information for classification. This might result in even better accuracies.

In short, integrating frequency information in few-shot learning has the potential to improve deep learning as a whole.

Acknowledgements

We would like to thank our expert reviewer Jiapan Guo and our peer reviewers Tom Couperus and Andra Trandafir for providing us with useful feedback that allowed us to improve the quality of the paper.

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[2] L. Bertinetto, J. F. Henriques, P. H. Torr, and A. Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

[3] X. Chen and G. Wang. Few-shot learning by integrating spatial and frequency representation. *2021 18th Conference on Robots and Vision (CRV)*, pages 49–56, 2021.

[4] H. Cheng, S. Yang, J. T. Zhou, L. Guo, and B. Wen. Frequency guidance matters in few-shot learning. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11780–11790, 2023.

[5] J. Frank, T. Eisenhofer, L. Schonherr, A. Fischer, D. Kolossa, and T. Holz. Leveraging frequency analysis for deep fake image recognition. *International conference on machine learning*, page 3247–3258, 2018.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

[7] Y. Hu, V. Gripon, and S. Pateux. Leveraging the feature distribution in transfer-based few-shot learning. *arXiv preprint arXiv:2006.03806*, 2020.

[8] J. Huang, D. Guan, A. Xiao, and S. Lu. Fsdr: Frequency space domain randomization for domain generalization. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6887–6898, 2021.

[9] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[10] Q. Li, L. Shen, S. Guo, and Z. Lai. Wavelet integrated cnns for noise-robust image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7245–7254, 2020.

[11] X. Luo, J. Xu, and Z. Xu. Channel importance matters in few-shot image classification. *ArXiv*, abs/2206.08126, 2022.

[12] W. Ma, K. Li, and G. Wang. Location-aware box reasoning for anchor-based single-shot object detection. *IEEE Access 8*, pages 129300–129309, 2020.

[13] R. J. Majid Rabbani. An overview of the jpeg2000 still image compression standard. *Signal processing: Image communication 17.1*, pages 3–48, 2002.

[14] P. Mangla, N. Kumari, and et al. "charting the right manifold: Manifold mixup for few-shot learning. *WACV*, page 2218–2227, 2020.

[15] Y. Rao, W. Zhao, Z. Zhu, J. Lu, and J. Zhou. Global filter networks for image classification. *Advances in Neural Information Processing Systems*, page 34:980–993, 2021.

[16] U. Sajid, W. Ma, and G. Wang. Multi-resolution fusion and multi-scale input priors based crowd counting. *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5790–5797, 2020.

[17] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128:336 – 359, 2016.

[18] J. A. Stuchi, M. A. Angeloni, R. F. Pereira, L. Boccato, G. Folego, P. V. Prado, and R. R. Attux. Improving image classification with frequency domain layers for feature extraction. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2017.

[19] M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *Proceedings 1998 IEEE International Workshop on Content-Based Access of Image and Video Database*, pages 42–51. IEEE, 1998.

[20] O. Vinyals, C. Blundell, T. Lillicrap, and D. W. et al. Matching networks for one shot learning. *Advances in neural information processing systems,*, page 3630–3638, 2016.

[21] O. Vinyals, C. Blundell, T. P. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *Neural Information Processing Systems*, 2016.

[22] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. *The Caltech-UCSD Birds-200-2011 Dataset*. jul 2011.

[23] H. Wang, X. Wu, Z. Huang, and E. P. Xing. High-frequency component helps explain the generalization of convolutional neural networks. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[24] K. Xu, M. Qin, F. Sun, Y. Wang, Y. kuang Chen, and F. Ren. Learning in the frequency domain. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1737–1746, 2020.

[25] H.-J. Ye, H. Hu, D. chuan Zhan, and F. Sha. Few-shot learning via embedding adaptation with set-to-set functions. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8805–8814, 2018.

[26] X. Yue, Z. Zheng, S. Zhang, Y. Gao, T. Darrell, K. Keutzer, and A. S. Vincentelli. Prototypical cross-domain self-supervised learning for few-shot unsupervised domain adaptation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13829–13839, 2021.

[27] Y. Zhang, Z. Dong, L. Wu, and S. Wang. A hybrid method for mri brain image classification. *Expert Systems with Applications*, 38(8):10049–10053, 2011.

Visual analysis of disinformation: A comparative overview

Alexandra Stan and Robin Guichelaar

Abstract—The development of information technology has led to people relying on social media platforms to obtain information. User-generated content, often including visual elements such as images and videos, contributes to the spread of large volumes of unverified information. Visual elements impact the emotions and opinions of individuals while increasing the credibility of disinformation, resulting in the escalation of societal divisions. Machine learning models have been widely employed to analyze the contained textual information and manage the fast spread of disinformation, while methods incorporating visual elements remain less explored. In this paper, machine learning and deep learning methods of detecting false information incorporating visual elements are summarized and compared. The comparison is based on the degree of inclusion of the visual elements in the classification process and how it influences the performance of false information detection.
It is found that the best performance is achieved when the visual element is considered separately by analyzing its past context. The optimal results were obtained for a Random Forest model, which attained an accuracy of 97.8%, and a Long Short Term Memory Network model, with an accuracy of 99%.

Index Terms—disinformation detection, false information, social media, visual content, machine learning, deep learning

1 Introduction

In recent years, the increasing popularity of social media platforms has shaped the way in which people communicate with each other. Millions of users can contribute to the fast propagation of large volumes of unverified information. Taking advantage of the means provided by social media platforms, users with malicious intent spread fake news, which often results in other users unknowingly disseminating them. In this context, manual verification of the information found on these platforms becomes unfeasible, highlighting the need for an automatic machine learning approach that classifies content as real or fake.

Nowadays, individuals engage with social media content that offers a summarized version of news through easy to consume multimedia formats like images or videos. It has been shown that social media posts containing images obtain 18% more clicks, 89% more likes and 150% more retweets compared to those without images [4]. Moreover, visual content often increases the credibility of the news. The rapid spread of fake news can have significant detrimental effects at an individual level and in society at large. As an example, before the 2016 US presidential election campaign, Americans were exposed to an average of one to three fake stories from known publishers, which affected the election results [1]. Similarly, after the disappearance of the Malaysia Airlines flight MH370 in 2014, many fake images indicating that the plane was found became viral on social media platforms, causing emotional distress among the families of the passengers [3].

It is crucial to detect fake information before it spreads widely. False information detection is an active research field in the machine learning community, and numerous models in this domain have been proven effective in identifying online disinformation [8]. The majority of existing research focuses on textual content alone, overlooking the visual element and the consistency between text and image. This motivates the need for further research on machine learning methods that also analyze the visual elements.

The present work provides an overview of three papers proposing different machine learning and deep learning approaches to the problem of detecting false information incorporating visual elements on social media platforms. Our goal is to thoroughly analyze and compare the methods proposed by Boididou et al. [3], Varshney and Vishwakarma [12], and Liang et al. [11] based on the degree of inclusion of the visual elements in the classification process and how it influences the performance of false information detection. Moreover, our second objective is to assess whether deep learning models outperform machine learning models in terms of performance.

Background information related to the existing research efforts on false information detection is provided in Section 2. Section 3 presents the three approaches we aim to compare and their respective results, which are discussed in Section 4. Finally, Section 5 summarizes and concludes the discussion, while Section 6 outlines possible directions for future work.

- *Alexandra Stan is with the Faculty of Science and Engineering, University of Groningen, E-mail: a.stan.7@student.rug.nl.*
- *Robin Guichelaar is with the Faculty of Science and Engineering, University of Groningen, E-mail: r.guichelaar@student.rug.nl.*

2 Background

A social media post containing visual elements is considered misleading if it fails to accurately represent the referenced event. Boididou et al. [3] define three types of misleading multimedia content: visual content from a past event used to describe a current event, intentionally manipulated visual content through techniques like tampering or photoshopping, and visual content posted with a false claim about the depicted event.

In recent years, extensive research has been done on unimodal false information detection methods, i.e. methods that only looked at textual content. Numerous papers looked at deep learning models to classify text-based information as real or fake. The majority of these models are based on Convolutional Neural Networks (CNNs) [13, 7], Long Short Term Memory Network (LSTM) [13, 6], Recurrent Neural Networks (RNNs) [6], and Bidirectional Encoder Representations from Transformers (BERT) [13]. However, all of these methods would fail to detect false information in social media posts that use visual content from a past event to describe a current event, since the text is perfectly credible.

Multimodal approaches have been much less explored. Machine learning-based approaches [3, 12] mainly use classifiers such as Random Forest (RF) and Logistic Regression (LR). Although they obtain good results, their shortcoming is manual feature engineering [8]. This has lead to the recent shift to deep learning methods, which are mostly based on Bidirectional LSTM (Bi-LSTM) [12], CNNs and BERT [11]. While deep learning models provide automatic feature learning and improved performance, they are accompanied by high computational cost and pose challenges in interpretation. In this work, we aim to investigate the effectiveness of deep learning methods over their machine learning counterparts.

3 Methods and Results

This section starts by describing the dataset used for the comparison of the three different methods proposed by Boididou et al. [3], Varshney and Vishwakarma [12], and Liang et al. [11] in Section 3.1. The false information detection methods are presented in Sections 3.2–3.4, highlighting details in their specific approach to the problem such as modifications made to the dataset, classification algorithms or combinations of networks employed, manner in which the features were extracted, and procedures used for performance evaluation. Furthermore, we also present the outcomes of the three methods, outlining the quantitative results obtained in the different evaluation procedures that are specific to each study. Therefore, each section presenting one of the methods is split in two subsections describing its methodology and results. Section 4 provides a synthesis of the outcomes that facilitates the comparison between the methods.

3.1 Dataset

For a fair comparison, we investigated papers that employ the same dataset for false information detection. The dataset used to compare the performance of the methods proposed by Boididou et al. [3], Varshney and Vishwakarma [12], and Liang et al. [11] is the Verifying Multimedia Use 2015 (VMU) described in [2]. VMU is a widely employed dataset for verifying the veracity of multimedia content. It consists of social media posts on widely known events or news stories on the Twitter platform. The veracity of each multimedia event was manually verified by cross-checking with online articles and blogs. In total, there are 6225 real tweets posted by 5895 unique users and 9404 fake tweets posted by 9025 unique users. VMU contains 193 cases of real images and 218 fake images.

3.2 Analyzing tweet- and user-based features via agreement based classification

The method proposed by Boididou et al. [3] verifies the veracity of a given post by taking features extracted from the tweet and the user who published it and combining them in a two-step classification model. The authors opted to focus on tweet- and user-based features for the analysis, noting that the misleading visual element of a post may not contain any detectable traces of tampering, or even be untampered. Therefore, each visual element from a post was treated as an URL that points to an online article that reported on it. Boididou et al. stated that articles originate from reputable news providers that have sufficiently justified their decision regarding the veracity of the visual elements. This made it possible to assign ground truth labels to tweets containing visual elements. The proposed framework is illustrated in Fig. 1.

Changes to dataset The authors cleaned the VMU dataset by eliminating re-tweets and manually filtering out posts with humorous content or indications that their content is fake.

3.2.1 Method

The features extracted from the tweet in order to be fed into the two-step classification process are of two types: user- and tweet-based, described below.

User-based (UB) features Two types of features related to the Twitter account of the user who posted the tweet were considered: user-specific and link-based, which are also seen in tweet-based features. A selection of examples is displayed in Table 1. User-specific features refer to the profile characteristics of the user, for example number of followers and tweets, or whether or not they have a profile picture. Link-based features give information about the trustworthiness of the links present in the tweet. This information is quantified through Web metrics like the WOT score, which evaluates the credibility of the tweet via a crowdsourcing approach, and the in-degree and harmonic centrality, which both rely on the connections of the web graph.

Table 1. Overview of user-based (UB) features

User-specific	Link-based (for UB and TB)
#friends	WOT score
#followers	in-degree centrality
follower-friend ratio	harmonic centrality
#tweets	alexa country rank
#media content	alexa delta rank
has profile picture	alexa popularity

Table 2. Overview of tweet-based (TB) features

Text-based	Language-specific	Twitter-specific
#words	#positive sentiment words	#retweets
length of text	#negative sentiment words	#hashtags
#question marks	#slangs	has external link
#exclamation marks	#nouns	#mentions
has "please"	contains 1st person pronoun	#URLs

Tweet-based (TB) features Four types of features related to tweets were considered: text-based, language-specific, Twitter-specific, and link-based. Some examples are shown in Table 2. Text-based features represent attributes extracted from the text of the tweet such as its length, number of words, stylistic characteristics such as number of question marks, or binary features that mark the presence or absence of specific words. Language-specific features like number of positive and negative sentiment words or slangs are extracted from a predefined set of languages. Twitter-specific features refer to details like the number of hashtags, mentions or retweets of the tweet.

Agreement-based classification In the first classification step, two independent classifiers CL_1 and CL_2 were built for the tweet- and user-based features, respectively. Bagging was used to increase the accuracy of the predictions, with $m = 9$ instances of each classifier being created and tested on m independent subsets of samples taken from the training set. The subsets contain an equal number of samples for each class. The final predictions of the classifiers were determined through majority voting. LR and RF of 100 trees were employed as classification algorithms.

To improve the model's generalization ability, the authors introduced agreement-based retraining as a second classification step. The strategy followed a semi-supervised learning paradigm: it divided the test set into the *agreed* and *disagreed* subsets by comparing the outputs of each of the m pairs of instances of CL_1 and CL_2. The samples in the *agreed* subset were given the label they agreed on, and subsequently used to retrain the best performing classifier between CL_1 and CL_2. The chosen classifier was employed to re-classify the samples in the *disagreed* subset. The authors investigated two retraining strategies: CL^{ag}, which only used the samples in the *agreed* subset to train the best performing classifier, and CL^{tot}, which used the entire training set extended with the samples in the *agreed* subset.

3.2.2 Results

The authors evaluated the contribution of bagging and agreement-based retraining to the accuracy of the framework. To assess the performance of bagging, the classifiers CL_1 and CL_2 were implemented both with and without this method. For the latter case, an equal number of random fake and real samples were taken from the training set. The use of bagging led to an improvement of approximately 10% for CL_1 and 4% for CL_2. Boididou et al. tested the agreement-based retraining approach by comparing the performance of CL_1 and CL_2 individually, of their concatenation CL^{cat}, and of the two proposed models CL^{ag} and CL^{tot}. The evaluation was conducted using both the LR and RF classifiers.

The findings show that the agreement-based retraining variations

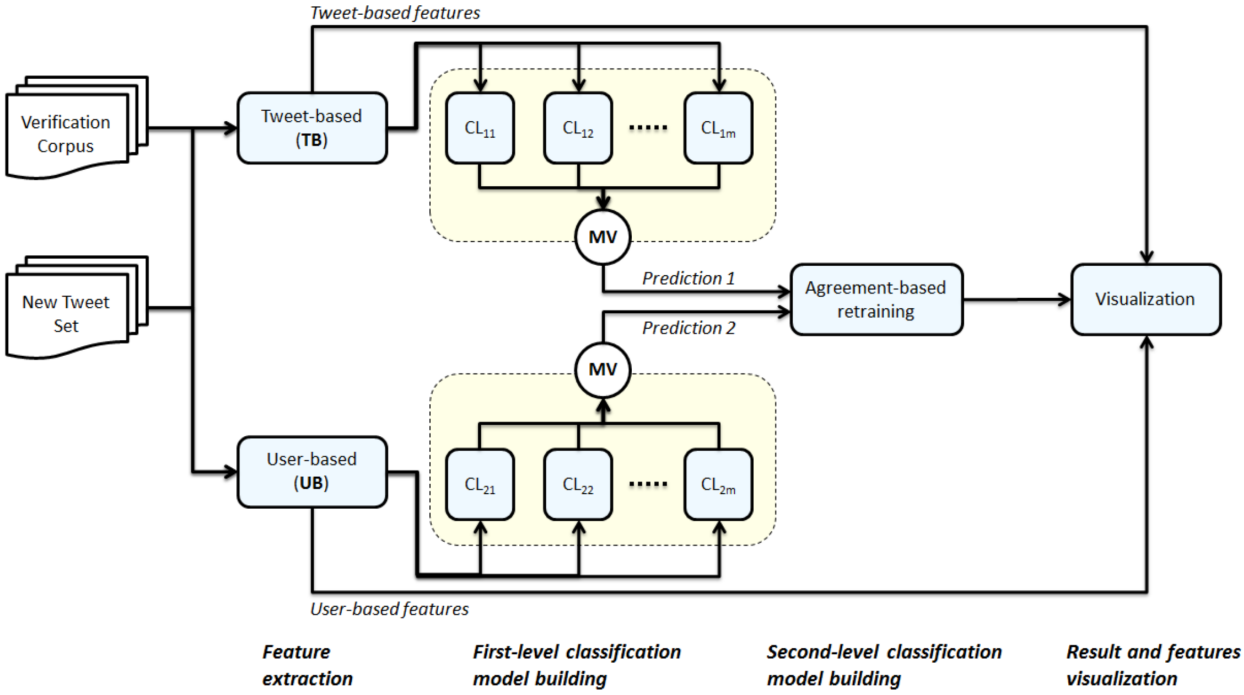


Fig. 1. Overview of framework proposed by Boididou et al. [3]. The TB and UB features were analysed separately by multiple classifiers CL_{in}, with $i \in \{1,2\}, n \in \{1,\ldots,m\}$, after which the final prediction was chosen through majority voting (MV). Samples on which the classifiers disagree were subjected to the Agreement-based retraining step.

perform better than CL_1 and CL_2. Compared to the best accuracy of CL_1, obtained in combination with RF, CL^{ag} and CL^{tot} resulted in around 1% improvement. Similarly, for CL_2, whose best accuracy was also obtained in combination with RF, CL^{ag} and CL^{tot} resulted in around 9% improvement. Although the agreement-based methods perform better overall, CL^{cat} using LR outperforms both CL^{ag} and CL^{tot} on the VMU dataset. Additionally, they found that LR performs better than the RF classifier overall. As for the two retraining approaches, CL^{ag} had a slightly better accuracy and F1 score than CL^{tot} on average, achieving the highest F1 score of 93.5% using RF. A notable observation is that, when paired with the LR classifier, CL^{tot} performed better than CL^{ag}, while CL^{ag} with RF outperforms CL^{tot}. The derived implication is that, to a certain extent, the agreement-based retraining approach is dependent on the classification algorithm.

3.3 Analyzing past context of images via machine learning and deep learning methods

Varshney and Vishwakarma [12] improve the previous method by incorporating features obtained from the image contained in the tweet in the prediction process. This was done via tracing the image on an image search engine and collecting information about its past context. The image search engine retrieved the top 10 titles found concerning the input image, which were subsequently used to extract features for the image. This approach is effective if the tweet whose credibility we want to assess contains an image only. The authors observed that the quality of the titles found for an image is one of the essential measures in improving the performance of the model. Therefore, two image search engines were employed and tested: Google Images and Bing Visual Search. As shown in Fig. 2, the classification model received two inputs: the combination of features extracted from the image and the text in the tweet, and only the features related to the image in the tweet. The performance of the proposed approach was tested using both machine learning and deep learning models. Furthermore, the authors performed a comparison with other state-of-the-art methods on the VMU dataset, among which is the method proposed by Boididou et al. [3].

Changes to dataset The main focus of this paper was to analyze Twitter posts containing text and images. Therefore, the authors filtered out the posts containing videos.

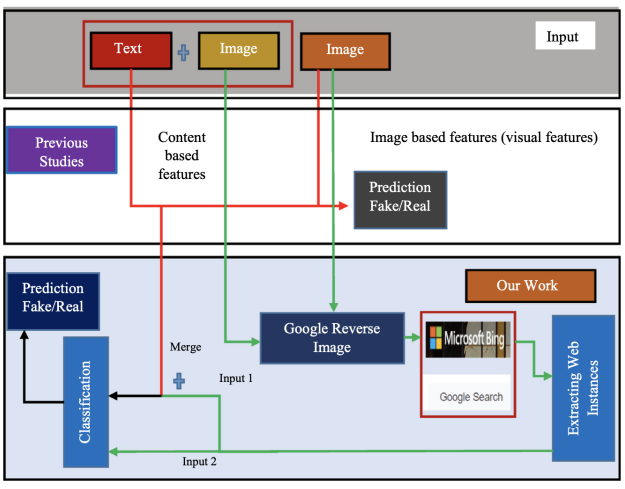


Fig. 2. Overview of framework proposed by Varshney and Vishwakarma [12], where the text and image were analyzed together. In addition, the image was separately investigated by reverse image searching, after which extracted features were merged and classified.

3.3.1 Method of machine learning model

To verify the veracity of a given input using a machine learning model, the authors created a set of five features: two trace of doubt features, two trace of fake features, and semantic similarity. The features were

defined for the text of the tweet, referred to as the query, and for the titles associated with an image.

Trace of doubt When the user expresses uncertainty regarding the query or the image contained in the tweet, the value of the trace of doubt feature is set to 1, otherwise it is set to 0. To identify the tweets in which the user expresses doubt, a corpus of phrases indicating uncertainty was compiled from the dataset. This corpus includes expressions like "is it", "is that", "not sure", and "?", among others.

Trace of fake In the same way, the trace of fake feature was defined for cases in which the user does not support the claim in the query or in the image contained in the tweet. The corpus of phrases expressing opposition contained words like "false", "misleading", "scam", "forged", etc. If any of the phrases in the corpus is present in the query or in the titles associated with the image, the value of this feature is 1, otherwise 0.

Semantic similarity To determine the contextual similarity between the query and the titles associated to the image contained in a tweet, the semantic similarity score has been computed using the semantic-text-similarity Python library. The similarity scores given for a set of sentences range from 0 to 5. A threshold value of 1.3, found empirically, was used to identify whether the query and titles are contradictory or not. The semantic similarity score between the query and the image was found to be an effective feature in classifying the tweet as fake or real.

The features presented above were given as input to the machine learning model, the performance of which was evaluated using the following classification algorithms: RF, LR, Naive Bayes, Linear SVM and K-Nearest Neighbor.

3.3.2 Results of machine learning model

The authors found that the RF and Linear SVM outperform the other classifiers, both obtaining the highest accuracy of 97.81% and F1 score of 97.8% using the Bing Visual Search engine. In comparison, as outlined in Section 3.2.2, Boididou et al. achieved the highest F1 score of 93.5% for the combination of their agreement-based method CL^{ag} and RF.

3.3.3 Method of deep learning model

The Bi-LSTM has been employed to extract the complex hidden representation from the query and the image associated to the tweet. Bi-LSTM is a type of RNN that captures the past and future context of the input by processing it in both forward and backward directions simultaneously. LSTM learns long-term dependencies in sequential data by employing memory cells and gating mechanisms to selectively store and discard information as time progresses. Bi-LSTM avoids the problem of vanishing gradient, which is commonly linked to RNNs, preventing the model from learning useless dependencies between similar words.

Feature extraction via Bi-LSTM The query and the titles associated to the image contained in the tweet were preprocessed by removing stop words, URLs and punctuation, stemming and lemmatization. A space-separated padded sequence of words was created from the query and the titles corresponding to the image, which was then split into tokens. Each token was represented by a real number through the use of one-hot vector encoding embedding. The resulting vectors were then given as input to the Bi-LSTM to extract complex hidden features. The training, validation and test set were constructed by partitioning the data in the vectors. The iterative training process aimed to minimize the binary cross-entropy loss by leveraging the Adam optimizer.

3.3.4 Results of deep learning model

Evaluating the performance of the deep learning model using the Bing Image Search engine, it obtained an accuracy and F1 score of 99% and an almost negligible loss. As in the machine learning model, the Bing Image Search engine provided better accuracy results for the deep learning model than Google Images.

3.4 Refining and fusing text and image features via Text-CNN and SE

Liang et al. [11] adopted a similar approach by extracting features from the images contained in the tweet. The proposed deep learning model extracted features from the text and the image and fused them using a multimodal version of Squeeze-and-Excitation Networks (SE). At the same time, both the text and image features were enhanced via Text-CNN. The input of the classification model was the concatenation of the locally enhanced text and image features and the fused features. The authors performed a series of ablation experiments to validate the effectiveness of each module in their proposed framework.

Changes to dataset In this study, the authors only looked at Twitter posts containing both text and images. Posts containing only images or text, images used as gifs, black or white images or videos were filtered out. Moreover, posts containing multiple images were split into separate posts containing only one image.

3.4.1 Method

As illustrated in Fig. 3, the proposed framework consisted of four stages: text and image extraction, text and image feature fusion, and the classification process.

Text feature extraction The text features were extracted through the combination of BERT [5] and Text-CNN. BERT is a Transformer-based model, i.e. it processes sequential data such as text, predicting missing words within sentences. Compared to previous models, BERT can capture contextual information and understand language nuances more effectively by analyzing both the preceding and succeeding words. It achieves a high performance using a limited amount of training resources. However, it presents a slight deficiency in the treatment of local features. In order to extract features with more local information, the model proposed in this paper applied three layers of Text-CNN to process the text features extracted by BERT.

Image feature extraction The image features were extracted through the combination of the Swin Transformer (SWTR) and Text-CNN. SWTR captures both global and local features within an image through a self-attention mechanism. While traditional transformers used in computer vision tasks process the image patch by patch, SWTR performs hierarchical feature extraction through a shifted windowing mechanism, reducing computational effort and memory requirements. SWTR exhibits the same shortcoming in the quality of the extracted local features as BERT, and thus its output was further processed by three scales, or layers, of Text-CNN.

Feature fusion The authors employed the SE module [9] to fuse the text and image features extracted by BERT and SWTR, respectively. The SE module enhances feature representation in CNNs through the squeeze and excitation operations. During the squeeze operation, the feature maps are aggregated across their spatial dimensions into a single channel descriptor vector through average pooling [9]. In the excitation stage, in order to capture dependencies and correlations among the channels, the vector is fed into a two-layer feed-forward neural network that produces a vector of weights representing the importance of each channel. The SE module was chosen in order to mitigate the effect of noise and information loss during the fusion process that was seen in models based on Transformer, RNNs or CNNs. Liang et al. modify the SE module to obtain multimodal features following the fusion process of the text and image features. The resulting model was termed MSE and works by first computing the attention score of text features on image features and vice versa. The dot product of the scores and the corresponding text or image features was then calculated, with the result being given as input to the activation function. Finally, the output features were fed into an average pooling layer. Before the classification process, the locally enhanced text and image features and the features extracted by the MSE module were concatenated.

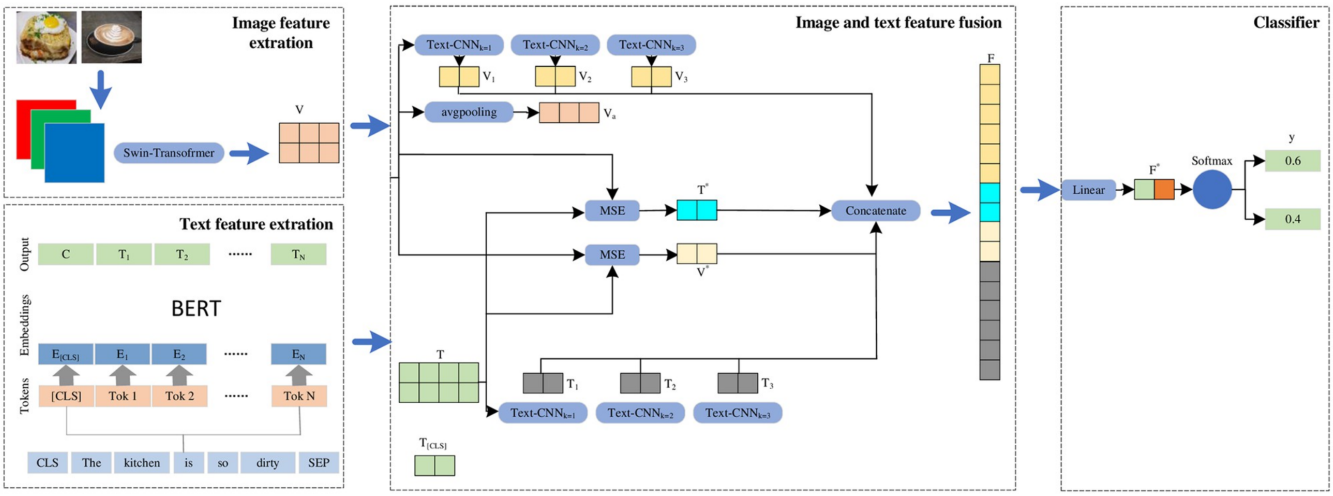


Fig. 3. Overview of framework proposed by Liang et al. [11], where image and text features are extracted separately, preprocessed by three layers of Text-CNN, concatenated, and then classified.

Classification The concatenated features are given as input to a fully connected layer which was followed by a Softmax layer. The cross-entropy loss function was employed to train the classifier. The authors did not mention which classification algorithms were used.

3.4.2 Results

The authors tested the effectiveness of using the proposed multimodal framework for false information detection over unimodal frameworks, as well as over other baseline multimodal frameworks. The same experimental conditions were used for each experiment. Furthermore, they also performed a series of ablation experiments on a different dataset [10] to validate the effectiveness of each module in their proposed framework.

Compared to unimodal models and the baseline models, the results show that the proposed framework outperforms both, achieving an accuracy score of 90.3%.

The ablation experiments are performed on the Weibo dataset [10]. The first ablation experiment removed the MSE and Text-CNN modules in order to demonstrate their effectiveness in the performance of the proposed method. The accuracy drops by 0.4% after removing the MSE module, by 0.8% when Text-CNN is removed, and by 1% when both are removed.

The second ablation experiment showed that the enhancements of the SWTR and BERT models with the three scales of Text-CNN resulted in accuracy improvements of 1.1% and 1.6%, respectively.

The third ablation experiment tested the benefit of adding exactly three different scales of Text-CNN to process the features extracted by BERT and SWTR. The accuracy of the model improved gradually as more Text-CNN of different scales were added. However, when adding more than three scales, the accuracy started to decrease.

The effectiveness of the proposed fusion method was tested by comparing the performance of the model with other different fusion methods. The fusion methods used for this comparison are based on feature summation, concatenation and other processing techniques using Transformer layers. The reader is referred to the paper by Liang et al. [11] for a detailed description of each. The results showed that the proposed MSE fusion had better results and used a much smaller number of parameters than the rest of the fusion methods.

4 Discussion

As highlighted by the results presented above, both machine learning and deep learning approaches have very good performance in detecting false information on the VMU dataset. For both types of approaches, the optimal performance was obtained for the method proposed by Varshney and Vishwakarma [12]. In their method, the image contained in the tweet and its past context were analyzed separately, and the features extracted were subsequently combined with the ones obtained by jointly analyzing the text and the image in the tweet. We believe that these results stem from the fact that this method introduces a more human-like approach by verifying the validity of an image using the internet and by analyzing the consistency between the text and the image.

All three studies modified the dataset by various preprocessing methods. The model proposed by Liang et al. [11] only works for tweets containing both text and images and presents experiments performed on a different dataset. Moreover, the methods did not all employ the same evaluation measures. Due to these reasons, a ranking of the three models summarized in this paper in terms of performance would be unfair.

5 Conclusion

We presented a thorough overview of three machine learning and deep learning methods proposed by Boididou et al. [3], Varshney and Vishwakarma [12], and Liang et al. [11], comparing their effectiveness based on how the degree of inclusion of the visual elements in the classification process influences the performance of false information detection. The best performance was achieved by analyzing the image contained in the tweet and its past online context separately, combining the extracted features with the ones obtained through a joint analysis of the text and the image in the tweet. Moreover, we have seen that deep learning approaches have better performance than machine learning methods while also avoiding the problem of manual feature engineering. However, it was impossible to quantitatively compare the performance of the three proposed models, as they preprocessed the dataset specifically to their use case.

6 Future work

As we have seen, incorporating multimedia content into the classification process to detect disinformation greatly improves the performance of the models and extends the applicability to more vast use cases. Future work could be directed at understanding the relationships between multiple modalities [8]. Currently, there are few datasets containing labeled multimodal data [4]. The performance of multimodal detection models can be further increased by verifying larger and more diverse cases, motivating the need for the creation of such datasets. Closely related to this is the time-consuming task of manual labelling of fake content, encouraging the investigation of false information detection with few or no labels given for training.

Early detection of fake content is paramount in order to prevent the fast spread of disinformation on online platforms. More research could

be aimed at informing the users of fake content at an early stage by signaling them during the diffusion process [8]. Currently, the detection task is seen as binary classification problem, but real-world data can display different levels of falseness. Further research could focus on splitting the false content into several categories with different degrees of deception [8].

ACKNOWLEDGEMENTS

The authors wish to thank the University of Groningen, anonymous reviewers, our friends in the study landscape, and the expert reviewer Jiapan Guo.

REFERENCES

[1] H. Allcott and M. Gentzkow. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–236, 2017.

[2] C. Boididou, S. Papadopoulos, D. T. Dang Nguyen, G. Boato, M. Riegler, A. Petlund, and I. Kompatsiaris. Verifying multimedia use at MediaEval 2016. October 2016.

[3] C. Boididou, S. Papadopoulos, M. Zampoglou, L. Apostolidis, O. Papadopoulou, and Y. Kompatsiaris. Detection and visualization of misleading content on Twitter. *International Journal of Multimedia Information Retrieval*, 7:71–86, 2018.

[4] J. Cao, P. Qi, Q. Sheng, T. Yang, J. Guo, and J. Li. *Exploring the Role of Visual Content in Fake News Detection*, pages 141–161. June 2020.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[6] S. Girgis, E. Amer, and M. Gadallah. Deep learning algorithms for detecting fake news in online text. November 2018.

[7] M. H. Goldani, S. Momtazi, and R. Safabakhsh. Detecting fake news with capsule neural networks, 2020.

[8] S. Hangloo and B. Arora. Combating multimodal fake news on social media: methods, datasets, and future perspective. *Multimedia Systems*, 28:1–32, July 2022.

[9] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu. Squeeze-and-excitation networks, 2019.

[10] Z. Jin, J. Cao, H. Guo, Y. Zhang, and J. Luo. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 795–816, New York, NY, USA, 2017. Association for Computing Machinery.

[11] Y. Liang, T. Tohti, and A. Hamdulla. Multimodal false information detection method based on text-cnn and se module. *PLOS ONE*, 17(11):e0277463–e0277463, 2022.

[12] D. Varshney and D. Vishwakarma. A unified approach of detecting misleading images via tracing its instances on web and analyzing its past context for the verification of multimedia content. *International Journal of Multimedia Information Retrieval*, 11:1–15, September 2022.

[13] A. Wani, I. Joshi, S. Khandve, V. Wagh, and R. Joshi. *Evaluating Deep Learning Approaches for Covid19 Fake News Detection*, page 153–163. Springer International Publishing, 2021.

Analyzing the Onboarding Experience of Students in Computing Science at the University of Groningen

Dogukan Tuna and Kaitlin Vos

Abstract—This study explores the onboarding process—defined as the method by which new interns are integrated into a company and prepared to contribute—to uncover its significance in ensuring interns' rapid adjustment and productivity, with the focus being on Computing Science students at the University of Groningen. Students at this university undergo an internship as part of their curriculum, but certain caveats exist in this procedure. The research highlights the challenges of streamlining this process to reduce the time at which code changes are reviewed and merged into their destination branch or the speed at which a new intern can start contributing meaningful code. By conducting an extensive survey among students who have completed internships at a variety of companies, from global tech giants to local startups, and either individually or in a group of students, we investigate the efficacy and impact of onboarding on interns' capacity to contribute significantly to the codebase. Anticipated results include the identification of factors leading to successful onboarding which could be shown by e.g. a speed-up in time to first commit, commonalities among diverse company types, inefficiencies within the onboarding process, and practical recommendations to enhance both onboarding efficiency and the code review process. This study aims to improve the internship experience for future students and, by increasing the efficiency of code reviews, to provide benefits to educational institutions and the tech sector at large.

Index Terms—Onboarding, code review, group dynamics, internships, students.

1 Introduction

Code review, as a term, describes a systematic evaluation of code that aims to detect flaws, improve code quality, and facilitate developers' understanding of the source code [11, 18]. The software industry aims to continuously improve this activity and, ultimately, the software development process to provide its consumers with a new or updated product faster [13, 12]. A wide range of literature covers possible factors that influence this process, such as influences of the code review cycle [4]. Interns at companies are one of the actors that can actively contribute to a company's code base and, thus, can impact the code review process.

Internships are an integral method to allow university students to gain real-world experience and a profitable learning opportunity next to their studies. At the University of Groningen, in the Computing Science degree, this is possible through our curriculum courses, such as Software Engineering. At the start of their internships, the students have to navigate through the onboarding process. Less attention has been paid to the impact of onboarding processes, especially distinguishing between individual and group onboarding experiences for interns.

The onboarding process refers to the process that helps interns or new hires learn the knowledge, skills, and behaviors necessary to succeed in their new organizations in order to become effective members [1]. However, the onboarding is difficult since it takes time before new staff gets up to speed [20]. One method to evaluate the efficiency of onboarding is by using the time until the first commit. The hypothesis is that the faster the first commit takes place, the more effective the onboarding is.

In this paper, we investigate the onboarding process of internships conducted at the University of Groningen, focusing on the Computing Science degree. We conducted a survey that asked the students about their onboarding process. By identifying common challenges, we will show how code reviews reflect the status of onboarding. The time window, therefore, extends from the first moment of contact to the completion of the onboarding. We address the following research questions:

RQ1. What factors of onboarding influence effective integration, and how do they differ between individual and group onboarding experiences?

RQ2. How do we make onboarding more efficient to reduce the time until the first commit for both individual and group learning internships?

RQ3. What specific steps can students and companies take during onboarding to maximize their code review efficiency?

In this paper, we summarize the related work in Section 2 and explain the methodology for the survey in Section 3. We present our findings in Section 4 and discuss the limitations of our approach in Section 5. Finally, we conclude the paper in Section 6.

2 Related Work

Previous studies have extensively examined the performance of recent graduates in industry settings. Since interns in our setting are below graduate level, the deficits graduates encounter are likely to be amplified.

Begel and Simon found that recent graduates mainly struggled in five areas: communication, collaboration, technicality, cognition, and orientation [3]. Radermacher and Walia [16] identified the most common knowledge deficiencies of graduating students, such as written and oral communication. Challenges included slow and inadequate responses, knowledge gaps due to poor documentation and missing developers, prolonged review times, limited testing tools, information overload, and isolation within companies. Additionally, graduates often felt compelled to solve problems independently to impress, hindering progress. The literature has not extensively covered how the nuances of onboarding—particularly the differences between individual and group settings—affect interns' ability to overcome these challenges and contribute to more efficient code reviews. Furthermore, literature mainly focused on recent graduates, rather than current students.

Sim and Holt [19] conducted their research on an exploratory case study with new team members and also found that administrative and environmental issues majorly contributed to new hire frustrations

- *Dogukan Tuna, E-mail: d.tuna@rug.nl.*
- *Kaitlin Vos, E-mail: k.x.vos@rug.nl.*

impacting their progress. They also discovered that the lack of documentation forces the new recruits to rely on mentors, who are usually already assigned to them due to the mentoring system. One of the responsibilities of a mentor, according to Sim and Holt, is to introduce the new hire to the team and provide them with the ability to reach out to the appropriate consultant when necessary. They identify the minimization of frustration as the most important objective to improve the onboarding of new hires. Ostroff and Kozlowski [15] observed that new hires with mentors also have a better understanding of organizational issues and practices, further highlighting the importance of the mentor system.

According to Dagenais et al.'s suggestions, additional studies have reported factors that affect the integration of new members, such as early experimentation, internalizing structures and cultures, and progress validation [7]. By assigning small, product-related tasks, the newcomers become familiar with the project, which acts as a preparation for the increase in the complexity of their tasks. By internalizing the structures and cultures of the domain and context by means of communication with the team and documentation, new hires can efficiently navigate and significantly contribute to it. Frequent progress validation ensures increased efficiency for newcomers and could possibly prevent them from getting stuck through early detection.

Our study seeks to build on these foundations by focusing on the specific aspects of individual vs group onboarding processes. By doing so, we aim to fill a gap in the literature regarding how onboarding influences interns' learning curves and their contributions to software projects. This focus allows for optimizing onboarding processes to improve the code review process, ultimately contributing to effective onboarding.

3 Methodology

In this section, we lay out the qualitative data collection methodology for our research. We constructed a survey consisting of 38 questions covering multiple sections. Each section aimed at specific pieces of information, which we describe further in this section. The sequence of the groups of questions followed the usual timeline of an internship in order to reinforce their recall abilities. Majority of the questions were open questions to allow the participants to supply additional information, if desired.

We determined and reached the target audience in a controlled manner by considering the academic years 2022-2023 and 2023-2024. We reached out through e-mail to a limited selection of those who have completed group internships during their Software Engineering course in the academic year 2022-2023. We also reached out to those who have completed an internship during their minor in the academic year 2023-2024. This came out to a total of 33 prospective participants who had been given 7 days to submit their responses.

The selection procedure of the internships conducted for Software Engineering starts with a survey to assess their knowledge, experience and skills, as well as, their projects of choice. Based on this information, the course organization constructs the teams for each internship.

We describe the design of the survey below:

Administration This selection of questions is targeted at collecting background knowledge about the internship. To get a better picture of the overall internship setting, we need to know whether the internship was completed alone and what part of the system has been worked on. It is also relevant whether they had to sign any contracts or NDAs, whether they were assigned a mentor, and whether they were supplied with any equipment from the internship company.

Group specific The Software Engineering course is in the second semester of the second year of the Computing Science degree, and it is the first course in which students have to work in groups and complete a 6-month project together. The students assign roles within the group, ranging from group leader to technical lead and documentation officer. The internships that are part of curricula minors take place in the first half of the third year, and is usually a full-time internship. With this group of open questions, we ask the students about the impact of the group dynamic on onboarding. The initial roles and tasks of the students are determined, and the evolution of the group dynamic is investigated, such as adaptability time w.r.t. changing/rotating roles and tasks.

Expectations and experiences The target audience of the survey is, generally, second- and third-year students at our university who are below the graduate level. Due to their inexperience, we tried to determine the degree of the knowledge deficit and its impact on the code review process. Factors such as expected knowledge, domain knowledge, and the scope of the project were taken into account. We further asked whether the complexity of their tasks increased and whether they were able to see the impact of their contributions during the internship.

First commit to codebase We investigate the interns' experiences leading up to the first commit to the codebase. We ask about the general timeline, if they were able to comply with it, and if any problems arose. Furthermore, we question the feedback they received in these code reviews in order to get a better understanding of whether they met the expectations of the company in their first commit.

Further commits to codebase Upon further integration into the company, we once again investigate the commits and code reviews, this time taking their gained knowledge and experience into account. We ask if the timeline was easier to plan and comply with, whether their code review feedback differed at all, and whether they were easier to incorporate.

Opinions In this section, we ask for the students opinion on what could be improved on the future students' side and the internship company's side. These questions were open-ended to allow the student the freedom to answer in any size they wanted. We believe that this information could potentially help future interns increase their efficiency in code reviews.

3.1 Response processing

After collecting the responses, we processed them. Since most questions were open questions, we generalized them by means of common themes. The participant has control over the response to the open-questions and we, therefore, differentiate between the following categories regarding quality and content:

- Theme: the response is categorized into one of the belonging themes based on keywords used (e.g. when asked about the duration: "1-2 weeks" is categorized as weeks and "a couple days" is categorized as days)
- Ambiguous: the response is ambiguous and cannot be clearly categorized
- NA (not applicable): based on the response of (a) previous question(s), the question is not applicable
- Missing: the question is applicable but the participant did not enter a response

Some of the ambiguous responses could be assigned a theme based on the answer to a follow-up question that provided context.

4 Results and Discussion

Among the targeted Computing Science students at the University of Groningen, 67% participated in the survey. Table 1 summarizes the composition of the responses. The majority of the participants completed their internship in groups, and most internship projects focused on backend development.

Table 1. Composition of responses

Type	Number of Students
Alone	6
Group	16
Total	22

4.1 RQ1: Factors influencing integration

Figure 1 indicates a difference in the time to first commit between individual and group onboarding experiences. Individual internships range from a duration in terms of days to a duration in terms of months, where the majority committed their first change in days. This suggests a quicker adaptation and integration into the coding tasks. Those who worked in a group needed at least a duration in terms of weeks to commit to the code base.

A potential reason for this occurrence, that is also consistent with the literature, is the fact that the students do not possess the required project management skills. This difference could worsen the problems with communication and teamwork that have been identified in existing literature, such as the works of Hagan et al. [9] and Brechner [6], highlighting the deficiency in project management skills. Besides the lack of soft skills, students also face difficulty writing an integrated piece of a larger project while their peers simultaneously write their pieces [10].

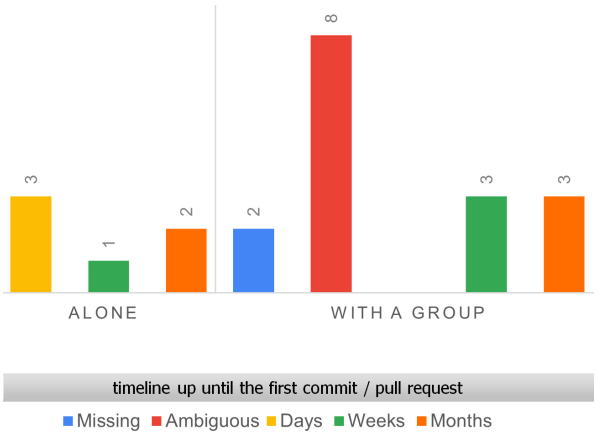


Fig. 1. Duration in terms of days, weeks and months until first commit as a function of alone vs group

Notably, the integration during the onboarding of groups into the company is mainly limited to project needs. Only a very few are invited to the company's communication platform and company meetings, limiting the exposure to broader company operations. The methods of integration are:

1. Company meetings, such as daily stand-ups or weekly company team meetings.
2. Company communication platform, such as Slack or Microsoft teams.
3. Project needs, which entails tailoring the project's needs with minimal integration into the company.

The students who completed their internship individually have all been part of the company meetings, which can be observed in Figures 2 and 3. This confirms our previous suspicion of the individuals being integrated more effectively. This allows to explore whether groups being integrated with the same treatment will lead to a shorter time period until the first coding contribution.

Fig. 2. Method of integration (ambiguous and project needs) and duration until first commit in terms of days, weeks and months as a function of alone vs group

Fig. 3. Method of integration (company communication platform and company meetings) and duration until first commit in terms of days, weeks and months as a function of alone vs group

Figure 4 summarizes the accessibility of the company supervisor to questions and the corresponding duration of the first commit. We can see that the majority of mentors were available to answer questions daily. This includes meetings, e-mails, and messages on the company communication platform. Commits produced in days have only been achieved through the daily accessibility of the corresponding mentor.

Rastogi et al. [17] observed that it was helpful for new hires to spend more time with the manager and team during the first two months to reduce the time until becoming effective members of the team, and that newcomers who are not assigned a mentor experience a significant loss of time. Since the Software Engineering course and the individual internships last an entire semester of $\sim 5-6$ months, time is of essence.

Berlin and Robin [5] also note the cost of mentorship but still see their applicability in many domains. They observed that incidental learning is a benefit of interactions with the expert since they go beyond the answer to the question by including history, facts, assumptions, and strategies. To reduce the expert's time, the apprentices in their study tried to understand the problem and narrow it down to something concrete. They also multi-tasked as they faced obstacles and noted them down as preparation for their next meeting with the expert. By providing the interns with multiple directions and isolated aspects to be explored, the internship companies could try to optimize the mentorship system that is often a central part of the onboarding process. Begel and Simon [2] further note that a slightly more experienced colleague mentoring new hires could be beneficial since they are more likely to be able to relate to them than a much more experienced colleague. This could also facilitate the project management skills that most students lack since slightly more experienced colleagues have had to learn this more recently.

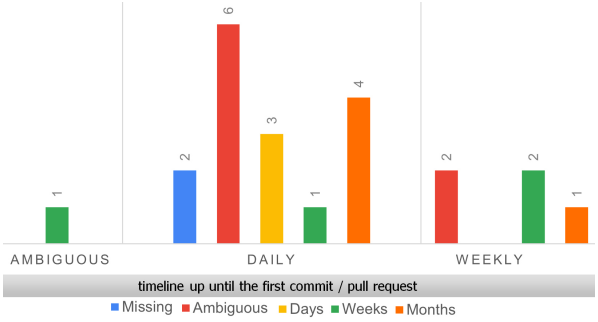


Fig. 4. Duration until first commit as a function of the availability of company supervisor

4.2 RQ2: Increase efficiency of onboarding until first commit

Through our analysis of intern feedback, two important sources of wasted time have been identified. According to the feedback the students receive on the first commit, the first source of wasted time occurs during the timeline up until that point. In 20% of the cases, the interns believe that the feedback could have been avoided. This can be seen in Figure 6. The students' access to more information could have prevented the majority of these situations. This information does not only include technical information, including in-house coding conventions, but also knowledge bases and where to find them. This also aligns with the suggestions the students provided regarding the improvement of the onboarding process. Access to these resources could mitigate this wasted time.

Dagenais et al. [7] observed that the developer-oriented documentation often fails to include the appropriate level of detail, and without its existence, newcomers have to rely on other methods to obtain this information, which usually involves trial and error. Documents are also stored in different places, in different formats, and are stale, which contributes to this time loss [17]. The subjects in Begel and Simon's study [3] also faced this, but they recognized that documentation becomes stale quickly. They instead desired information on people so that they could reach out to the correct person with the appropriate responsibilities for their issues. Since the development process might be too complex to document, people are, in their opinion, the most valuable documentation resource. Since our students went through similar issues, it might be worth exploring the effects of interns being informed about the people they can reach out to, which could benefit the integration of the interns as well, and evaluating whether this affects the code processes.

It is not only important for interns that the documentation is present and fairly up-to-date but also that it is centralized and clearly communicated to accelerate the code review process.

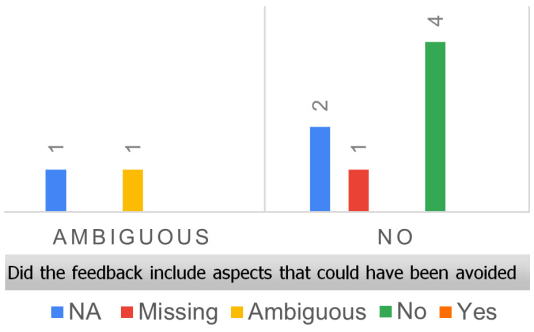


Fig. 5. Whether it was helpful to the student as a function of whether the company supervisor reviewed the code

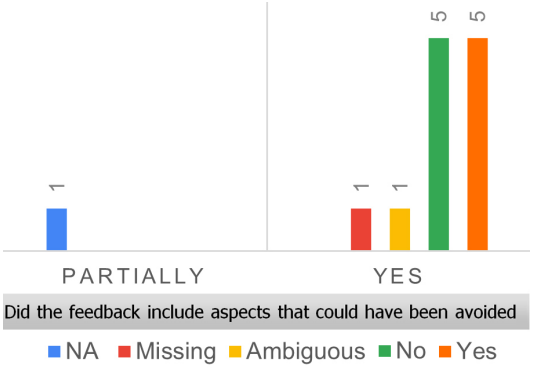


Fig. 6. Whether it was helpful to the student as a function of whether the company supervisor reviewed the code

The second source of frustration is the requirement definition and the project organization. Figure 7 shows that projects with a change in timeline can be perceived as problematic by the students. The students whose project timeline changed are mainly responsible for the suggestion of more frequent meetings with the company supervisor. More frequent meetings could maintain clarity and project alignment for a smoother onboarding process. Meetings where newcomers can ask questions and get proactive suggestions based on their progress reports were very helpful, according to Dagenais et al. [7], since not all forms of meetings are helpful to recently hired members.

Rastogi et al. [17] observed that well-chosen starting tasks that give a complete overview of the system reduce the time for a new hire to reach the productivity level of the team, which can help to mitigate the source of frustration. However, it is also important for students to be able to experience "real customers" to resemble the commercial software industry instead of very clear requirements and expectations, according to Brechner [6]. According to Begel and Simon [2], this might help close the gap between academia and industry. Students should be able to offer insights to their internship company and conduct usability studies. Understanding what the client truly needs is essential to good product design and execution. One could almost consider this point expressed by the students as a positive outcome since this effectively simulates the industry, potentially, at the cost of slower code reviews. This is, therefore, a trade-off. On one hand, it is important to teach the students the necessary skills in university as preparation for the industry, but on the other hand, it is important to mitigate their frustrations to optimize the code review process.

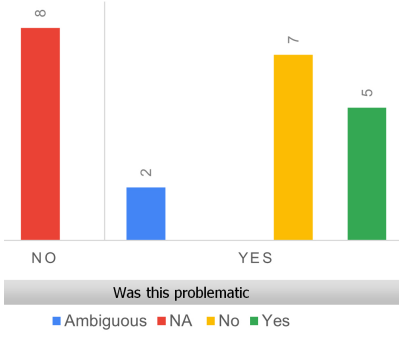


Fig. 7. Whether that was problematic as a function of whether the project timeline deviated

One of the frequently recommended suggestions of the students to improve the efficiency is to start the onboarding process sooner. The administration of the onboarding often includes an NDA, of our participants more than 70%, as can be seen in Figures 8 and 9. Equipments

and code is only received after the NDAs have been signed and processed. This is a source of delay regarding the internships and delays the date the interns can start the actual internship, potentially influencing the code review process. Around 50% of the students have reported that the process of the NDA took at least weeks. It is important to explore how this can be optimized, from either university and/or student side but also the company side concerning internal processes.

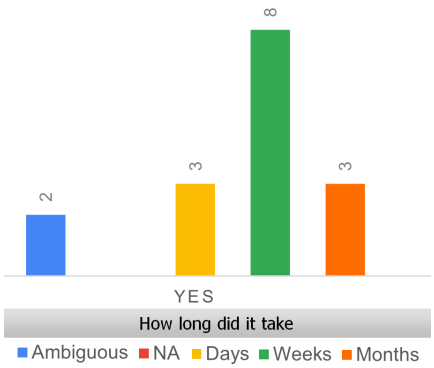


Fig. 8. Duration needed for the NDA as a function of whether they had to sign one

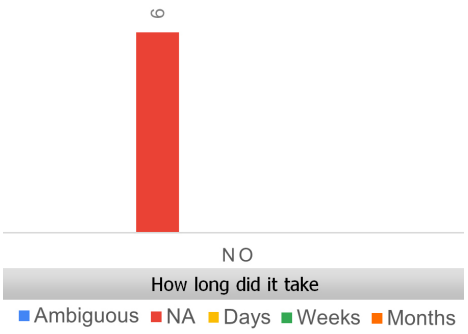


Fig. 9. Duration needed for the NDA as a function of whether they had to sign one

4.3 RQ3: Practical suggestions for all parties involved

1. **Integration of teams**: We have observed that only individuals have been able to contribute to the codebase in a matter of days and noticed clear discrepancies between the integration of individuals and group internships. Fisher [8] shows that social support also reduces stress and facilitates positive adjustment outcomes for new hires. This leads to the hypotheses that if groups were integrated as effectively, whether they could match this speed. An interesting option to explore would be to invite the interns to work on-site in close vicinity with the mentor and the company team. Previous studies have shown that a hands-on approach at the start of the internship has led to many successful outcomes.

2. **Selection and administration procedure of interns**: Only a small number of students possessed the required knowledge for their internship. This suggests that the way the students are selected for, e.g., the group internships for the Software Engineering course focuses on the potential of developing these skills by having skills in similar areas. This ensures that the students are able to do their internship without learning an entirely different domain but still enforce the learning of new skills or extending them. After the selection, the usually lengthy NDA process starts. The company could clearly communicate the amount of time they need to sort this internally and the required documents, so that the university can set clear deadlines, allow the students to get the required paperwork ready and possibly start the selection procedure sooner.

3. **Internship project structure**: Hagen et al. [9] reported that employers are dissatisfied with recent graduates' project management skills. By modularizing the internship projects for both individuals and groups could help the students prioritize easier since they do not have to dissect one big task with complicated dependencies. This could also levitate the lacking teamwork skills since responsibilities can be more easily delegated and dependencies are minimized and contribute to the optimization of meetings with the mentor by facilitating multi-tasking.

4. **Proactivity of students**: The most frequent advice included in the responses is to be proactive. This can come in many forms, such as the following mentioned in the responses: asking questions, asking for feedback, and proposing your own ideas. The company, however, has the authority to enforce this. Dagenais et al. [7] observed that to better integrate new hires, the team also has to be proactive by answering questions, frequently checking in on progress, assigning a mentor, and walking the new hires through their first tasks through guided exploration. Waiting for new hires to come up with questions to ask has the opposite effect. This observation is similar to that of Major and Kozlowski [14].

5 Limitations and Future Work

Participation pool We collected data from 33 prospective participants, of whom 67% responded from the Computing Science degree at the University of Groningen. Due to the size and locality of the participation pool, the results might not be generalizable.

Survey The final survey has undergone three rounds of reviews by a Software Engineering professor and two peers within the time span of a week, and due to the high international rate of the Computing Science degree at the University of Groningen, some questions may have been susceptible to ambiguous interpretations, potentially impacting response consistency. To mitigate this issue, more evaluation rounds are necessary and the survey could be supervised, so that the participant can ask questions to which they will receive an immediate clarifying response.

Processing of data The processing of responses to open-ended questions involved subjective interpretation to appropriately categorize them. As the survey targeted students, responses often lacked formal sentence structures and were provided in an informal manner. This may have introduced some level of interpretation bias and resulted in many responses being assigned to either "ambiguous" or "missing". In order to prevent this, many of the open questions should be replaced by more well-defined questions, such as multiple-choice questions and Likert scales, to better control the variety of the responses.

Timing Many participants provided estimates or failed to recall specific details due to the delay between the survey and their internship experiences. To enhance response accuracy, future surveys could prompt students to track their internship experiences quantitatively and qualitatively, allowing for more precise and reliable data collection rather than relying solely on memory.

6 Conclusion

This study explored the onboarding process in industry settings, particularly focusing on how it impacts interns' code reviews. Through a survey targeting students from the University of Groningen who have undergone internships in various companies, we gained valuable insights into the factors that influence onboarding efficiency and, consequently, code reviews.

Our findings underscore the necessity of a well-structured onboarding process for potentially accelerating interns' adjustment periods and enhancing their ability to contribute meaningfully to the codebase. Notably, the presence of a mentor, access to comprehensive documentation, and integration into the company's culture emerged as major determinants of a successful onboarding experience. Individual internships resulted in faster first commits compared to group settings, indicating that team dynamics may impact onboarding speed.

The implications of these insights extend to educational institutions, companies, and interns alike. For companies, refining the onboarding process not only boosts the code review process but also enhances overall productivity and intern satisfaction. Educational institutions, on the other hand, can leverage these findings to tailor their curricula and preparation programs, better aligning them with the realities of the tech industry. Interns, equipped with a clearer understanding of what to expect and how to navigate their onboarding, can adjust their strategies to maximize their learning and contribution.

The study acknowledges limitations such as the limited number of participants and the use of self-reported data, which require additional investigation. In light of the potential threats to the internal and construct validity of our research, we suggest many directions for further studies. To improve the external validity of the findings and gain a more thorough understanding of the onboarding process in various industries, it would be beneficial to include interns from a broader range of disciplines and firms in the participant base. In addition, by including quantitative measures of the time to first commit, lines of code, or commit frequency, it is possible to reduce the limitations of relying on self-reported data. This approach would provide a stronger framework for assessing the impact of onboarding practices on intern productivity.

Although our research primarily targeted Computing Science students at the University of Groningen, we believe that our findings have the potential to provide significant insights in other contexts as well. Nevertheless, we see this claim as a possible challenge to the generalizability of our research. Internship protocols and customs can vary considerably depending on the field, sector, and geographical area. Considering this, we advise against blindly applying our findings to all internship environments without careful evaluation. Future study should aim to confirm and expand upon our findings in various settings, ensuring that the recommendations for improving the onboarding process may be applied more widely. Conducting comparative research with participants from different backgrounds and organizational settings would be highly beneficial for evaluating the universality of our findings and discovering specific contextual elements that may impact the success of onboarding techniques.

To conclude, we highlight the crucial impact that the onboarding procedure has on interns' code reviews in professional settings. Stakeholders can greatly improve the internship experience by addressing the identified difficulties and utilizing the insights gathered. This will have a positive knock-on effect on the efficiency and innovation capability of the tech industry as a whole. Future research within this field may yield even greater onboarding process optimization, which in turn may facilitate an easier transfer for interns and increase their value to their home organizations.

REFERENCES

[1] T. Bauer and B. Erdogan. Organizational socialization: The effective onboarding of new employees. *APA handbook of industrial and organizational psychology*, 3:51–64, 01 2011.

[2] A. Begel and B. Simon. Novice software developers, all over again. In *Proceedings of the Fourth International Workshop on Computing Education Research*, ICER '08, page 3–14, New York, NY, USA, 2008. Association for Computing Machinery.

[3] A. Begel and B. Simon. Struggles of new college graduates in their first software development job. *SIGCSE Bull.*, 40(1):226–230, mar 2008.

[4] M. Beller, A. Bacchelli, A. Zaidman, and E. Juergens. Modern code reviews in open-source projects: which problems do they fix? In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, page 202–211, New York, NY, USA, 2014. Association for Computing Machinery.

[5] L. M. Berlin and R. Jeffries. Consultants and apprentices: observations about learning and collaborative problem solving. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 130–137, 1992.

[6] E. Brechner. Things they would not teach me of in college: what microsoft developers learn later. In *Companion of the 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, OOPSLA '03, page 134–136, New York, NY, USA, 2003. Association for Computing Machinery.

[7] B. Dagenais, H. Ossher, R. K. E. Bellamy, M. P. Robillard, and J. P. de Vries. Moving into a new software project landscape. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, page 275–284, New York, NY, USA, 2010. Association for Computing Machinery.

[8] C. D. Fisher. Social support and adjustment to work: A longitudinal study. *Journal of management*, 11(3):39–53, 1985.

[9] D. Hagan. Employer satisfaction with ict graduates. In *Proceedings of the Sixth Australasian Conference on Computing Education-Volume 30*, pages 119–123. Citeseer, 2004.

[10] B. Kitchenham, D. Budgen, P. Brereton, and P. Woodall. An investigation of software engineering curricula. *Journal of Systems and Software*, 74(3):325–335, 2005.

[11] O. Kononenko, O. Baysal, and M. W. Godfrey. Code review quality: how developers see it. In *Proceedings of the 38th International Conference on Software Engineering*, ICSE '16, page 1028–1038, New York, NY, USA, 2016. Association for Computing Machinery.

[12] G. Kudrjavets. *The Need for Speed: Increasing the Code Review Velocity*. PhD thesis, University of Groningen, 2023.

[13] C. Maddila, S. S. Upadrasta, C. Bansal, N. Nagappan, G. Gousios, and A. van Deursen. Nudge: Accelerating overdue pull requests toward completion. *ACM Trans. Softw. Eng. Methodol.*, 32(2), mar 2023.

[14] D. A. Major, S. W. Kozlowski, G. T. Chao, and P. D. Gardner. A longitudinal investigation of newcomer expectations, early socialization outcomes, and the moderating effects of role development factors. *Journal of applied psychology*, 80(3):418, 1995.

[15] C. Ostroff and S. W. Kozlowski. The role of mentoring in the information gathering processes of newcomers during early organizational socialization. *Journal of Vocational Behavior*, 42(2):170–183, 1993.

[16] A. Radermacher and G. Walia. Gaps between industry expectations and the abilities of graduates. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, page 525–530, New York, NY, USA, 2013. Association for Computing Machinery.

[17] A. Rastogi, S. Thummalapenta, T. Zimmermann, N. Nagappan, and J. Czerwonka. Ramp-up journey of new hires: Tug of war of aids and impediments. In *2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–10, 2015.

[18] T. Savor, M. Douglas, M. Gentili, L. Williams, K. Beck, and M. Stumm. Continuous deployment at facebook and oanda. In *Proceedings of the 38th International Conference on Software Engineering Companion*, ICSE '16, page 21–30, New York, NY, USA, 2016. Association for Computing Machinery.

[19] S. Sim and R. Holt. The ramp-up problem in software projects: a case study of how software immigrants naturalize. In *Proceedings of the 20th International Conference on Software Engineering*, pages 361–370, 1998.

[20] M. Stein and L. Christiansen. *Successful Onboarding (PB)*. McGraw Hill LLC, 2010.

Sentiment Analysis for Cost Management Actions Related To Changes In Infrastructure as Code (IaC) Artifacts

M. al Shakoush, V. Andrikopoulos, D. Feitosa

Abstract—A popular approach for automating the management of IT infrastructure, and specifically of cloud infrastructure is the use of Infrastructure-as-Code tools. These tools offer a number of benefits but as the managed infrastructure grows they introduces challenges towards managing their related artifacts, especially in relation to the cost of operating this infrastructure. Analyzing the commit messages associated with changes to such artifacts to understand the sentiments expressed by developers in them is therefore an important step towards being able to predict appropriate actions. To this effect, in this work we present an exploratory study assessing to what extent existing approaches on sentiment analysis can be used for this purpose, before looking into developing a fine-tuned model. We find that while general-purpose sentiment analysis models interpret IaC-related commit messages poorly, Large Language Models like GPT 3.5 show superior performance. However, fine-tuning LLMs for specific domains presents challenges. These insights suggest LLMs' potential in IaC management and inform future research directions.

Index Terms—sentiment analysis, Infrastructure as code, large language models, cost, cloud infrastructure

1 Introduction

Infrastructure as Code (IaC), an approach to the management of IT infrastructure, has recently gained widespread acceptance. According to Gartner , by 2025 more than 80% organizations will adopt IaC as a fundamental practice to manage IT infrastructure. IaC treats cloud infrastructure configuration files as software code to be interpreted by automation tools. This allows organizations to manage and provision their infrastructure through machine-readable definition files (rather than going through manual configuration), automate software deployments, and apply version control techniques to configuration files and other IaC artifacts [27]. Popular IaC tools include Terraform , AWS CloudFormation , and Azure Resource Manager .

As infrastructure grows in complexity, managing changes to IaC artifacts becomes more challenging. Understanding how these changes impact the cost of infrastructure management in the cloud is therefore crucial . This is because despite IaC's several benefits, cost management can be a significant concern for organizations and a lack of insights into the cost implications of changes to IaC artifacts can result inunexpected expenses .

Towards this goal, this work explores whether sentiment analysis can be used to enable organizations to effectively manage their cloud infrastructure costs. Sentiment analysis is an appropriate tool for this task since it can identify opinion polarity, emotions, and attitudes in text data, with several applications in diverse domains [23, 3, 5, 21]. Yet, the potential of sentiment analysis for cost management in IaC remains under-explored [10]. To this effect, this work seeks *to explore the potential of sentiment analysis for understanding cost management actions in response to changes in IaC artifacts*. These changes could be related to cost optimization, performance improvement, or other actions taken by application developers. By analyzing the sentiments associated with the commit messages implementing those actions, we hope to be able to (eventually) identify patterns and trends that can help us predict cost management actions related to these changes.

The literature already contains several works applying sentiment analysis to commit messages (e.g. of open source projects on GitHub [7, 23, 13]), to produce customized models that "understand" technical jargon [2, 1, 4, 13, 30]. To the extent of our knowledge, there has been no existing approach in the field that aims at IaC artifacts. Thus, it makes sense to start our investigation by assessing existing sentiment analysis approaches, both general-purpose and Software Engineering-oriented, fare on a dataset of commit messages on IaC artifacts. Simultaneously, the advent of Large Language Models (LLMs) and their text-processing capabilities offer an additional opportunity to use them for the same task.

This paper's contribution is threefold. Firstly, it comes with a first of its kind, manually labeled dataset of the perceived sentiments of commit messages for IaC artifacts. Secondly, it examines whether classical machine learning-based and current LLM-based (both un- and fine-tuned) models can be used to identify the sentiment for each commit message. Finally, it provides a comparative evaluation of the efficacy of those different approaches.

The paper is structured as follows. Section 2 discusses the study design, including the dataset choice and adopted models for further investigation. Section 3 presents the investigation results. Section 4 discusses the interpretation and implications of these results for practitioners and researchers. Section 5 summarizes related work, addresses threats to validity. Lastly, Section 6 provides conclusions and future work.

2 Study Design

In this section, we discuss the specific research questions defined for this work and detail our approachto answering them. In the following section we discuss our choice of dataset used as input, the selected models, and the process for analyzing the results.

2.1 Objective and RQs

This study focuses on examining the relationship between sentiment and cost management actions in IaC, and the effect of LLMs on this task. Consequently, our research questions are:

RQ1. To what extent is it possible to use existing sentiment analysis models from the literature to establish this relation?

RQ2. How efficient is it to use existing pre-trained LLMs for the same purpose?

RQ3. Can the performance be improved by fine-tuning LLMs?

To answer these questions we will be using as input a dataset compiled by Feitosa et al. [6], which is publicly available[1] and contains relevant data pertaining to IaC artifacts from commits and issues on GitHub. From that dataset we only keep the commit messages. To address these RQs, we initially use various state-of-the-art models from the literature to analyze this dataset. Subsequently, we apply a non-fine-tuned LLM for the same purpose, followed by fine-tuning a pre-trained LLM. Processing the dataset (separately) using all of the models yields a labeled dataset of what every model concludes the sentiment is based on the commit messages. To further analyze this labeled

[1] https://github.com/feitosa-daniel/cloud-cost-awareness/blob/main/dataset.json

data we employ different agreement metrics to assess the agreement levels among the models themselves. The ground truth for all cases is established by manually labeling the original dataset. In the following we go through the individual steps in more detail.

2.2 Dataset and Ground Truth

Feitosa et al. [6] conducted a mining operation of public GitHub repositories to investigate the prevalence of cost-related concerns in the development history of IaC artifacts. Specifically, the mining process was aimed at identifying repositories with a codebase that exceeded a specified percentage of IaC artifacts. Two sub-datasets were collected, namely 538 commits messages and 208 issues, both related to Terraform IaC files. This paper primarily focuses on the commits, which will serve as input for all models used to construct the labeled dataset. This approach simplifies the task by reducing complexity, since commit messages are single strings while issues can comprise multiple messages serving different purposes. However, investigating the connection between issues and pull requests can be advantageous in future work.

The labeling of this input dataset took place using a tripolar (i.e. positive, neutral, and negative) tagging method [29]. The tripolar tagging of sentiments is a fundamental design decision in this study. By moving beyond the binary sentiment classification, we aim to capture nuanced sentiments expressed in commit messages. For instance, a developer's sentiment towards a code change could be positive due to efficiency improvements, neutral for minor bug fixes, or negative for complex refactoring. The tripolar tagging approach enhances sentiment analysis granularity, making it more contextually relevant to software engineering tasks [26]. To ensure the accuracy and reliability of the tripolar tagging, we established ground truth through rigorous manual data labeling. To facilitate this, a web application was created, providing a user-friendly front end instead of working directly with the JSON file. To ensure the validity of the study, two of the authors of this paper independently labeled the dataset, resolving discrepancies through discussion. The aforementioned application is publicly available on GitHub. However, the URL is currently omitted for double-blind review purposes.

The application comprises a backend and a frontend component. The former reads the dataset and transmits the commit messages to the latter sequentially. The user selects the sentiment for each message, the frontend transmits it back to the backend, which writes the sentiment to the appropriate file. To expedite the development process, the technology stack consists of React as the frontend framework for its rapid development capability, and FastAPI for the backend, due to its speed and performance.

2.3 Models and Data Collection

Model Selection In sentiment analysis, language model selection is meticulously conducted, drawing from a diverse range documented in the literature. This includes both transformer-based architectures and prior machine-learning methodologies. The process in this work begins with deploying general-purpose sentiment analysis models, transitions to models tailored for software engineering applications, attempts to use a model not trained specifically for sentiment analysis, and culminates in developing a bespoke model tailored to the unique requirements of this study.

General purpose and Software Engineering-specific sentiment analysis models The Stanza [24], VADER [12], and TextBlob [9] models were selected for the first part of the study. These models are commonly employed in sentiment analysis and are thus chosen as the go-to models for the purpose of this study. They are also pre-LLM machine learning models, renowned for their interpretability, which adds to their appeal. As software engineering-specific models, we intended to use Senti-SE [22] and Senti-CR [1] due to their prior use in the context of software engineering topics. But, we encountered issues with both, that prevented us from deploying them to label our dataset.

Specifically, when attempting to utilize SentiSE, we discovered that the model was built using Java with Apache Ant as the building tool. Although the project repository includes a jar with dependencies[2] that should be executable, it does not function as expected due to missing classes. It was discovered that the structure of the code for the model needed fixing to build it locally, rather than relying on the included jar. Upon attempting to local building, it also became apparent that some crucial files, required for training the model, were missing. Despite contacting the author with a request to provide the missing files or further instructions on how to build the model, no response was received.

Senti-CR, built using Python, also proved challenging. The repository lacked clear indications of the Python version, the dependencies and their exact versions. This made getting started with the model very difficult. After using the latest version of Python and fixing all syntax errors, the necessary libraries were updated. Subsequently, the model was trained, taking several hours. However, after experimenting with the resulting model it appeared that contrary to what is discussed in [1], the model only produces bipolar instead of tripolar labels.

In both cases, we attempted to resolve those issues by initiating communication with the authors of the corresponding papers regarding the aforementioned concerns. Nevertheless, we were not able to get a response. As a result, we decided to skip using a software engineering trained model in this study.

Pre-trained LLM Next, we selected GPT 3.5, which uses the text-davinci-003 model, as our non-fine tuned LLM of choice. The selection of GPT 3.5 was based on its advanced natural language processing capabilities, which contribute to its proficiency in comprehending and evaluating intricate textual information. The capacity to produce coherent and contextually appropriate responses enhances sentiment analysis accuracy, particularly in subtle or uncertain contexts. Furthermore, GPT 3.5's extensive training on diverse datasets ensures a thorough understanding of various domains, including software engineering, rendering it a valuable asset for sentiment analysis within the field.

In order to obtain the sentiment from GPT, we utilized the OpenAI API to retrieve the sentiment and the reason using the following prompt:

```
response = openai.Completion.create(
    model="text-davinci-003", # GPT 3.5
    prompt= "Use JSON to format the response like this:
    { sentiment: "sentiment here", reason: "reason here" }.
    Classify the sentimentof the following sentence and
     give me the reason:
    <text>",
    temperature=0, max_tokens=60, top_p=1.0,
    frequency_penalty=0.0, presence_penalty=0.0
)
```

Given that we are utilizing GPT 3.5 purely for the output, we have not utilized either the few-shot or the zero-shot methods.

GPT-3.5's results were inadequate for direct use due to inconsistency in formatting. Specifically, the generated responses did not seem to adhere to the given prompt, requiring JSON formatting for the response, necessitating us to format it by hand. Moreover, additional processing was required to clean the data of special characters and extraneous text generated by GPT 3.5, including unsolicited text completion attempts and inconsistent results notation.

Fine-tuned LLM For producing a fine-tuned model, we used the Alpaca LLM as our base model, namely the Alpaca-Lora[3] variety which reconstructs the Stanford Alpaca using low-rank adaptation [11]. Alpaca was selected due to its expertise in domain-specific sentiment analysis, which proves highly advantageous in fields characterized by specialized vocabulary and contextual nuances, such as software engineering. Alpaca's capacity to undergo fine-tuning using specialized datasets enables it to adapt to the specific linguistic intricacies and technical terminology commonly encountered in software development conversations [28]. As such, Alpaca is optimal for accurately deciphering sentiments expressed in software engineering texts, where

[2] https://stackoverflow.com/questions/19150811/what-is-a-fat-jar

[3] https://github.com/tloen/alpaca-lora

conventional models may struggle with the unique terminology and contextual elements of the field. Moreover, Alpaca's fine-tuning potential allows for customization, catering to the specific requirements of individual projects, thereby enhancing its efficacy in targeted sentiment analysis tasks.

A fine-tuned version of Alpaca has already been created with sentiment analysis in mind . A dataset comprised of tweets pertaining to Bitcoin and their corresponding sentiments has also been used for this purpose . To assess potential improvements, we fine-tune the model on the combination of this dataset with the one used in our paper. To be more precise, we employ the following models for our experiments: Alpaca-LoRA-7B fine-tuned with our dataset only, Alpaca-LoRA-7B fine-tuned with our dataset and the Bitcoin dataset, and GPT 4 Alpaca 13B which comes already fine-tuned with the new GPT 4 instruction set.

For the fine-tuning of Alpaca, we used the provided script by Alpaca-LoRa after modifying the instruction and the output it to adhere to our dataset only and the combination of our dataset and the bitcoin sentiment dataset. Thus getting:

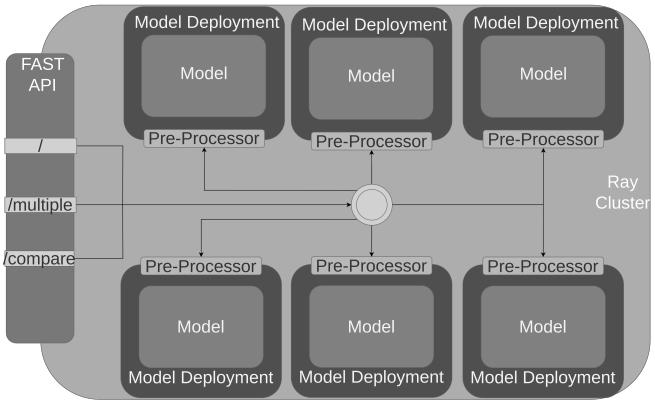


Fig. 1: Architecture of the sentiment analysis pipeline

```
"instruction": "What is the sentiment of the following sentence?",
"input": "<sentence>",
"output": "<expected sentiment>"
```

When generating the output for Alpaca we used the default values provided by Alpaca-LoRa.

```
f"{What is the sentiment of the following sentence?",
f"{data['commit_message']}",
temperature=0.1 top_p=0.5, top_k=40, nr_of_beams=4,
128,# Max tokens
```

We note that these models, including those fine-tuned with our dataset, also presented output issues similar to those encountered with GPT 3.5. Thus, we also solved the inconsistencies in formatting and unsolicited additional text by these models manually. One significant refinement is the translation of sentiments (e.g., 'disappointed' or 'frustrated') to our standard ('negative').

Data Collection Setup

To be able to label the dataset using all the selected models, we developed a horizontally scalable pipeline for automation. The architecture of which is depicted in Figure 1. Every model is containerized in its model deployment along with its pre-processor, allowing for tailored processing of incoming textual data. The deployments are hosted inside a `Ray cluster` facilitating easy parallelization and scaling of the load. A load balancer is integrated in front of all the model deployments to ensure even distribution of load across replicas. We exposed specific endpoints, using `Fast-API` , to invoke the methods in these models. Using these endpoints, a user can obtain sentiment from one or multiple models at a time, and compare the models. This pipeline is publicly available on GitHub. However, the URL is currently omitted for double-blind review purposes.

The pipeline was used to label our input datasets using the Stanza, VADER, TextBlob, and GPT 3.5 models. We utilized the "multiple" endpoint within our pipeline architecture, facilitating simultaneous sentiment evaluation from all integrated models in parallel. The Alpaca model was not incorporated directly into the pipeline due to its lengthy and resource-intensivefine-tuning process. Instead, we opted to host the model as an API using the `Gradio Client` , allowing for convenient access and the requests capabilities. In terms of implementation, integrating the Alpaca model into the pipeline would follow a similar procedure as adding the GPT model, requiring the inclusion of the API URL. After retrieving the sentiment from all Alpaca versions mentioned, we successfully completed the task, resulting in the acquisition of a newly labeled dataset.

2.4 Data analysis

To assess the models' effectiveness, we employ Cohen's kappa and Krippendorff's alpha as evaluation metrics, treating each model as an independent rater and looking for inter-rater agreement or reliability:

Cohen's kappa Inter-rater reliability assessment, critical for research validity, commonly employs the Kappa statistic to measure agreement among raters. Evaluating consistency across raters instills confidence in data accuracy. [18]. Cohen's Kappa coefficient ranges from $\kappa = 1$ for complete agreement to $\kappa = 0$, if there is no agreement among the raters other than that which would be expected by chance. The statistic can also be negative [25], by chance if there is no correlation between the ratings of the two raters, or if there is a tendency for the raters to give contrasting ratings.

Krippendorff's alpha Krippendorff's alpha is a reliability measure used to assess the agreement or reliability of data annotations or codings across multiple raters or coders. It is particularly applicable when dealing with nominal or ordinal categorical data, where raters classify items into discrete categories or assign ordinal values [8]. Negative α values indicate significant disagreement or systematic bias, while an α value of 0 suggests chance-level agreement. Positive values of α reflect higher levels of agreement and indicate a more reliable and consistent coding process. An α of 1 indicates perfect agreement, which is rarely achievable in real-world data coding tasks.

Use of the metrics These metrics provide measures of inter-rater agreement or reliability, allowing us to assess the accuracy of the models. The manual labeling of the data set serves as the ground truth. To provide a more complete picture of the results, we will also provide the precision, recall and f1-score of each model (calculated based on the ground truth). Depending on the research question being dealt with, the relevant metric is selected to compare different sets of models. Non-LLM based models will act as baselines for further comparisons with both the pre-trained Chat GPT and the different versions of the fine-tuned Alpaca models.

3 Results

Before looking at the results for individual RQs, we offer a comprehensive view of the new data. In Table 1 we provide the precision, recall and f1-score for each model (also measured against the ground truth). GPT 3.5 appears to be the overall best performing model and closest to the ground truth, with Alpaca 13B with GPT 4 in second. Here we highlight that while GPT 3.5 is a closed model, Alpaca is open, which can make it particularly compelling for future work exploration.

3.1 RQ1: Existing analysis models

General-purpose sentiment analysis models encountered difficulties comprehending technical terminology commonly used in the software industry and were prone to errors resulting from typographical mistakes. Furthermore, the degree of agreement between the three models applied to the labeled dataset was low, resulting in a 57% of conflicts between the models. More specifically, the calculated Krippendorff's alpha score of $\alpha = 0.15$ across the three models indicates very low

Table 1: Comparison of classifier models

Model	Precision	Recall	F1 Score
VADER	0.28	0.31	0.31
TextBlob	0.30	0.33	0.37
Stanza	0.20	0.31	0.34
GPT 3.5	**0.65**	**0.64**	**0.68**
Alpaca 7B	0.36	0.30	0.45
Alpaca 7B with Bitcoin	0.36	0.30	0.45
Alpaca 13B with GPT 4	0.51	0.61	0.48

agreement among raters or coders, practically indistinguishable from what would be expected by chance alone. Furthermore, examining the Cohen's kappa score range of 0.11–0.23 in the pairwise comparison between these models (i.e. without considering the ground truth) , confirms the low agreement between the models, similarly aligned with what would be expected by chance alone.

Including the ground truth in the comparison clearly demonstrates the unsuitability of these models for analyzing the input dataset. A maximum Cohen's kappa score of 0.16 and a Krippendorff's alpha score of 0.11 suggest virtually no agreement between the general-purpose models and the ground truth, beyond what would be expected by chance. Additionally, the precision, recall and F1-score are similarly low compared to the best-performing model (see Table 1).

The dataset used is particularly specific to IaC artifacts changes, containing very specific commit messages in relation to cloud computing terms (e.g., 'VM', 'Resources', 'Cluster') further demonstrating that these are not fit in the realm of IaC.

3.2 RQ2: The LLM (GPT 3.5)

As expected, GPT 3.5 demonstrates robustness towards typos, exhibiting less confusion compared to the other three models. Moreover, GPT 3.5 shows an understanding of sorts for technical jargon and acronyms. For example, it identified that the acronym 'VM' stands for 'Virtual Machine' and identified that the sentiment of obtaining cheaper Virtual Machines is positive without any prompting on our part.

Many of the responses generated by GPT 3.5 (text_davinci_003) suggest it is capable of understanding the context of a given sentence and providing relevant and reasonable responses. It can also seemingly accurately capture and express the tone and intent of the original commit message. A Cohen's kappa score of 0.82 suggests substantial agreement between ground truth and the GPT 3.5 model in assigning sentiment labels. This level of agreement is considered very good, indicating that the sentiment labels provided by the raters closely align with each other and demonstrate a high degree of concordance. Therefore, GPT 3.5 can be better relied upon as a tool for this sentiment analysis task. That said, we still see room for improvement when factoring in the precision, recall and f1-score (see Table 1).

Comparing with the general-purpose models, Figure 2 shows Cohen's kappa scores ranging from 0.09 to 0.16, indicating a very low agreement between GPT 3.5 and the general-purpose models. Additionally, Krippendorff's alpha of 0.11 further confirms the low level of agreement. These results align with those of RQ1, highlighting the strong alignment between GPT 3.5 and the ground truth.

3.3 RQ3: The fine-tuned models

As discussed in Section 2.3, we use the base Alpaca(-Lora) model fine tuned in three different ways:

Alpaca 7B fine-tuned with our dataset Compared to ground truth, we observed a kappa of 0.35 (Figure 3) and precision, recall and F1-score of 0.3, 0.45 and 0.36 respectively (see Table 1). These scores suggest an overall improved performance compared to the evaluated general-purpose models. However, this is still substantially worse compared to GPT 3.5, which needed no fine-tuning. Given that we do not know the data used to train GPT 3.5 and how it was fine-tuned for instructions, drawing a precise conclusion regarding the lower performance of our fine-tuned model is not straightforward. That said, looking at

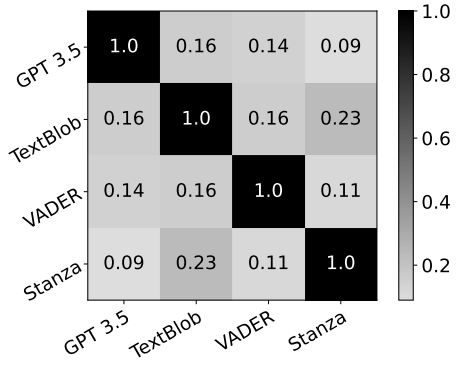


Fig. 2: Cohen's kappa — GPT 3.5 and general-purpose models (Krippendorff's alpha between these models is 0.11).

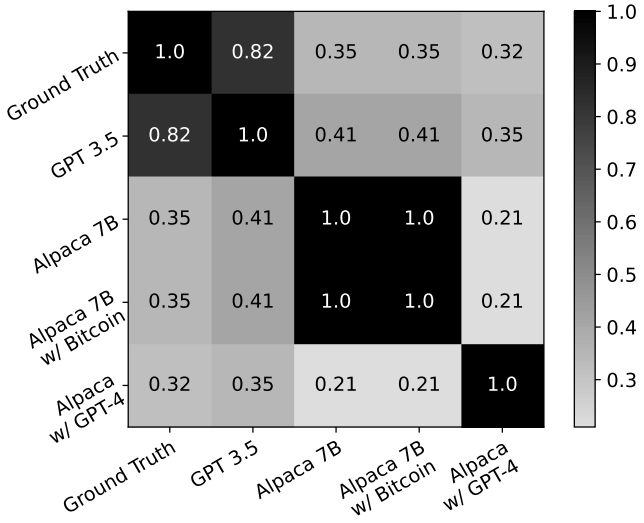


Fig. 3: Cohens's kappa with all fine-tuned LLMs (Krippendorff's alpha between these models is 0.42).

the agreements, reveals a higher agreement between Alpaca 7B and GPT 3.5 (i.e., $\kappa = 0.41$) compared to the ground truth. Furthermore, Krippendorff's alpha between the ground truth and two models is 0.53. This increased consensus between the fine-tuned model and GPT 3.5 suggest may suggest potential improvement through better instruction training.

Alpaca 7B fine-tuned with our and the bitcoin datasets Comparing the results of this model to the ground truth reveals the same scores as the previous model (i.e. Alpaca 7B without the bitcoin sentiments dataset). From Figure 3, it is also apparent that Alpaca 7B with and without the extended bitcoin dataset agree with a Cohen's kappa score of 1.0, i.e., a perfect agreement between two models. Altogether, we conclude that adding the bitcoin sentiments dataset does not affect the fine-tuning outcome.

Alpaca 13B fine-tuned with GPT 4 GPT 4 Alpaca 13B has a Cohen's kappa score of 0.32 with the ground truth and 0.35 with GPT 3.5 as indicated in Figure 3. While marginally worse than of the less complex Alpaca 7B models (fine-tuned with our dataset), we found the precision, recall and f1-score of Alpaca 13B to be substantially closer to those of GPT 3.5 (see Table 1). Notably, the recall is nearly identical, meaning the model is similarly able to make correct classifications. Upon inspection if the labels assigned by each model, the explanation becomes clear. The fine-tuned Alpaca 7B's do not classify any commit as neutral, and mark most commits with positive sentiment. This dataset imbalance contributes to higher kappa

scores for Alpaca 7B models. Meanwhile, Alpaca 13B's comprehensive performance, which accurately identifies neutral commits, contributes to superior precision, recall, and f1-score. Altogether, we position the Alpaca 13B model as second best in our list. We also note the Krippendorff's alpha between the ground truth, GTP 3.5 (the best model) and Alpaca 13B is 0.49, marginally similar to that between the ground truth, GTP 3.5 and the Alpaca 7B's. Whether this is because of the larger size of the LLM, or use of the GPT 4-instruction set is not clear. Interestingly, this version of Alpaca agrees more with GPT 3.5 than with the ground truth. However, we highlight a limitation of this model: in some instances, the parameters used - mainly the `max_tokens` - limited the output of the model. This would not have been a problem if Alpaca did not repeat the commit message in the output, resulting in the sentiment missing.

4 Discussion

4.1 Interpretation of Results

The study provides insights into the application of various sentiment analysis models for interpreting commit messages in Infrastructure as Code. A notable observation is the contrast in performance between general-purpose models, such as Stanza, VADER, and TextBlob, and Large Language Models like GPT 3.5. The latter demonstrates a significantly higher alignment with the manually established ground truth, highlighting the adeptness of LLMs in handling the complex vocabulary and nuances of the software engineering domain.

Interestingly, the fine-tuned versions of Alpaca LLMs displayed less surprising results. Both the Alpaca 7B model versions, with and without the Bitcoin dataset, showed only moderate agreement with the ground truth, and inferior precision, recall, and F1 scores compared to the other evaluated LLMs. This outcome suggests that fine-tuning LLMs for specific domains remains a complex task that does not always yield linear improvements in performance. The GPT 4-tuned Alpaca 13B model, however, did show improved results, emphasizing the potential benefits of deploying a larger model.

Furthermore, the discrepancies observed among the different models may corroborate the inherent challenge in sentiment analysis, which is particularly pronounced in software engineering, where technical jargon, sarcasm, and contextual references can significantly influence the sentiment conveyed in text. That said, the complexity of fine-tuning LLMs meaningfully is another factor to consider. Whether these observations generalize beyond the provided task and dataset is, however, the topic of future work.

5 Related Work

In this section, we review the literature and discuss the challenges of applying sentiment analysis in software engineering and the limitations of existing models.

The field of sentiment analysis in software engineering has utilized a range of models, from lexicon-based approaches to sophisticated machine learning and deep learning techniques. Early studies, such as those by Guzman et al. [7] and Pletea et al. [23], predominantly used lexicon-based models that relied on predefined lists of words with associated sentiment values. While effective for general sentiment analysis, these models often struggle with the nuances of software engineering language.

The main limitations in sentiment analysis within this domain stem from the challenge of interpreting technical jargon and informal language commonly found in software engineering texts. Where lexicon-based models can fail (e.g., with domain-specific terms and slang), machine learning models have the potential to be more robust. As the field progressed, researchers like Calefato et al. [3] and Jongeling et al. [14, 15] explored machine learning models. These models, including support vector machines and neural networks, offered improved accuracy by learning from data [21]. Jongeling et al. particularly employed machine learning to distinguish between sentiments in issue comments and stack overflow discussions [14] .

Nevertheless, machine learning models also require extensive, domain-specific training data, which can be scarce. Furthermore, these models grapple with the subjective nature of sentiments, which can vary greatly between individuals and contexts. Studies such as Lin et al. [17] and Novielli et al. [19] evaluated a number of state-of-the-art models' effectiveness in interpreting thecontext of software development, concluding that such models are not mature enough to be used in software engineering.

Obaidi et al.provides a comprehensive overview of the progression and application of various sentiment analysis methods within software engineering [20] , emphasizing the field's evolving needs. Similarly, Lin et al. highlights the broader scope of extracting sentiments and opinions, underscoring the importance of nuanced understanding in software development contexts [16] .

In attempts to address these limitations, researchers have proposed a number of sentiment analysis tools specially designed for software engineering. Islam and Zibra demonstrated that SentiStrength-SE, a version of SentiStrength fine-tuned for software engineering datasets, outperforms its predecessor, especially in distinguishing extreme sentiments from neutral ones [13]. More recently, Sun et al. proposed SESSION, a next step in SE-specific models that also leverages sentence structures [26] . The findings show noticeable improvements in comparison to past models.

In the wake of such advancements, it becomes evident that context is a key factor in sentiment analysis. Thus, investigating models relying on the transformer architecture becomes imperative. In this direction, Zhang et al. fine-tuned pre-trained transformer models (BERT, RoBERTa, XLNet, ALBERT) on a dataset comprising sentences from various SE sources such as StackOverflow posts and Jira issues. The results show that the transformer models outperform the state-of-the-art models by 6.5-35.6% in F1 scores [30].

Our work built on the findings of these studies. We started by exploring and checking the results of some of the cornerstone models in a new application domain (i.e., IaC-related changes). Then we delved into LLMs to further explore the potential of the context learned by these advanced transformer-based models, including fine-tuned ones, to deal with the intricacies of sentiment analysis in software engineering.

Recent advancements in the application of Large Language Models (LLMs) have opened new avenues for enhancing sentiment analysis within the software engineering domain. The study by Zhang et al. represents a significant step forward, demonstrating the potential of both smaller Large Language Models (sLLMs) and bigger Large Language Models (bLLMs) in tackling sentiment analysis challenges specific to software engineering [31]. Their empirical investigation, leveraging five established datasets, showcases the adaptability of bLLMs to the sentiment analysis for software engineering (SA4SE) task, even in scenarios marked by limited training data or imbalanced data distributions.

This exploration aligns with our research, underscoring the feasibility and efficiency of applying pre-trained LLMs, like GPT-3.5, to decipher the nuances of sentiments expressed in Infrastructure as Code (IaC) artifacts. Zhang et al.'s findings particularly highlight the edge of bLLMs in zero-shot and few-shot learning setups, a revelation that complements our study's examination of LLMs' capability in predicting cost-related actions from sentiments encoded in IaC artifact commit messages.

Furthermore, their comparative analysis between fine-tuned sLLMs and bLLMs underlines a critical insight pertinent to our work: the contextual sensitivity of LLMs to different types of sentiment analysis tasks within SE. Where bLLMs exhibit state-of-the-art performance on datasets with sparse or unevenly distributed training examples, sLLMs continue to thrive given sufficient and balanced training data. This distinction plays a pivotal role in our methodological choice and analytical framework, guiding our selection of GPT 3.5 for sentiment analysis within the IaC domain.

Zhang et al.'s research not only enriches the landscape of sentiment analysis in software engineering but also provides a foundational benchmark for our study. By leveraging the inherent strengths of LLMs, particularly in handling complex and domain-specific linguistic patterns, our research extends the conversation on the practical applications of sentiment analysis for cost management in cloud in-

frastructure development.

6 Conclusions & Future Work

This paper provides an exploration into the application of sentiment analysis on commit messages related to Infrastructure as Code (IaC) artifacts, specifically in the context of managing costs. Our research highlights that traditional, general purpose machine learning models, while useful, show limited efficacy in accurately interpreting the specialized and technical language of commit messages realated to IaC artifacts. Conversely, Large Language Models (LLMs), especially GPT 3.5, demonstrate superior performance, aligning more closely with the ground truth. This finding demonstrates the potential of LLMs in extracting nuanced sentiments from software engineering communications, a capability critical for understanding and predicting cost-related actions in IaC management.

Looking forward, the mixed results obtained from fine-tuning LLMs suggest that while they hold promise for enhanced domain-specific sentiment analysis, the process is intricate and does not consistently yield improvement. These insights pave the way for future research to refine sentiment analysis techniques in software engineering. Further research could focus on expanding dataset diversity, experimenting with varied models and fine-tuning methods, and integrating sentiment analysis tools into continuous integration and deployment workflows. This work, therefore, not only advances our understanding of sentiment analysis application in a novel context but also opens new avenues for research in the dynamic field of software engineering.

References

[1] T. Ahmed et al. Senticr: A customized sentiment analysis tool for code review interactions. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 106–111. IEEE, 2017. https://doi.org/10.1109/ASE.2017.8115623.

[2] E. Biswas, K. Vijay-Shanker, and L. Pollock. Exploring word embedding techniques to improve sentiment analysis of software engineering texts. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 68–78, 2019.

[3] F. Calefato et al. Sentiment polarity detection for software development. *Empirical Software Engineering*, 23(3):1352–1382, September 2017.

[4] J. Ding et al. Entity-level sentiment analysis of issue comments. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*, SEmotion '18, page 7–13, New York, NY, USA, 2018. Association for Computing Machinery.

[5] Y. K. Dwivedi et al. Setting the future of digital and social media marketing research: perspectives and research propositions. *Journal of Business Research*, 122:3–13, 2020.

[6] D. Feitosa et al. Mining for cost awareness in the infrastructure as code artifacts of cloud-based applications: an exploratory study, 2023.

[7] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: an empirical study. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, ICSE '14. ACM, May 2014.

[8] K. Gwet. On krippendorff's alpha coefficient. 10 2015.

[9] D. Hazarika et al. Sentiment analysis on twitter by using textblob for natural language processing. In *Proceedings of the International Conference on Research in Management & Technovation 2020*, ICITKM 2020. PTI, 2020.

[10] H. Hoffmann et al. Infrastructure as code: A systematic review of research challenges and opportunities. *ACM Computing Surveys (CSUR)*, 54(6):1–36, 2021.

[11] E. J. Hu et al. Lora: Low-rank adaptation of large language models, 2021.

[12] C. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. 01 2015.

[13] M. R. Islam and M. F. Zibran. Sentistrength-se: Exploiting domain specificity for improved sentiment analysis in software engineering text. *Journal of Systems and Software*, 145:125–146, November 2018.

[14] R. Jongeling, S. Datta, and A. Serebrenik. Choosing your weapons: On sentiment analysis tools for software engineering research. In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, September 2015.

[15] R. Jongeling et al. On negative results when using sentiment analysis tools for software engineering research. *Empirical Software Engineering*, 22(5):2543–2584, January 2017.

[16] B. Lin, N. Cassee, A. Serebrenik, G. Bavota, N. Novielli, and M. Lanza. Opinion mining for software development: A systematic literature review. *ACM Trans. Softw. Eng. Methodol.*, 31(3), mar 2022.

[17] B. Lin et al. Sentiment analysis for software engineering: how far can we go? In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18. ACM, May 2018.

[18] M. L. McHugh. Interrater reliability: The kappa statistic. *Biochem Med (Zagreb)*, 22(3):276–282, 2012.

[19] N. Novielli, D. Girardi, and F. Lanubile. A benchmark study on sentiment analysis for software engineering research. In *Proceedings of the 15th International Conference on Mining Software Repositories*, ICSE '18. ACM, May 2018.

[20] M. Obaidi and J. Klünder. Development and application of sentiment analysis tools in software engineering: A systematic literature review. In *Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering*, EASE '21, page 80–89, New York, NY, USA, 2021. Association for Computing Machinery.

[21] M. Obaidi and J. Klünder. Development and application of sentiment analysis tools in software engineering: A systematic literature review. In *Evaluation and Assessment in Software Engineering*, EASE 2021. ACM, June 2021.

[22] R. Paul, A. Bosu, and K. Z. Sultana. Expressions of sentiments during code reviews: Male vs. female. In *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, Feb. 2019.

[23] D. Pletea, B. Vasilescu, and A. Serebrenik. Security and emotion: sentiment analysis of security discussions on github. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, ICSE '14. ACM, May 2014.

[24] P. Qi et al. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020.

[25] J. Sim and C. C. Wright. The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements. *Physical Therapy*, 85(3):257–268, 03 2005.

[26] K. Sun et al. Exploiting the unique expression for improved sentiment analysis in software engineering text. In *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*. IEEE, May 2021.

[27] D. A. Tamburri et al. Tosca-based intent modelling: goal-modelling for infrastructure-as-code. *SICS Software-Intensive Cyber-Physical Systems*, 34(2):163–172, 2019.

[28] R. Taori et al. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[29] S. Vashishtha, V. Gupta, and M. Mittal. Sentiment analysis using fuzzy logic: A comprehensive literature review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(5):e1509, 2023.

[30] T. Zhang et al. Sentiment analysis for software engineering: How far can pre-trained transformer models go? In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, September 2020.

[31] T. Zhang, I. C. Irsan, and F. Thung. Revisiting sentiment analysis for software engineering in the era of large language models. https://synthical.com/article/c9e05c7a-fc63-4bc7-8758-cbf9a749d57b, 9 2023.

Impact of VR-Induced Nausea on VR Training Sessions for Professional UAV-Drone Operations

Minh Tam Le, and Himawan Indra Bayu

Abstract—In recent years, professionals have used Unmanned Aerial Vehicles (UAVs), also known as drones, to make their work easier, even though providing drones is not cheap. A proposed way to prevent costly failures when using drones as a supported tool is by introducing virtual training aimed at helping users familiarize themselves with drone environments. Several methods have been successfully implemented in the professional field. However, the potential for VR-induced nausea during extended VR training sessions needs to be considered. This raises the question of how to minimize these symptoms and ensure a safe and effective training experience. Our approach focuses on conducting a literature review concerning VR-induced nausea in VR training sessions for UAV-Drone operations. We contribute to the field by outlining design guidelines and recommendations for the development of VR training environments specifically tailored to Unmanned Aerial Vehicle (UAV) drone operations.

Index Terms—Virtual Reality (VR) Sickness, Virtual Reality (VR) Training Design, UAV-Drone Training, Causes of VR Sickness

1 Introduction

The rapid integration of Unmanned Aerial Vehicles (UAVs), commonly known as drones, across industries over the past decade has revolutionized task execution and data collection, leading to increased efficiency. This growth, however, necessitates a skilled workforce of UAV operators. In response, organizations are increasingly adopting VR training due to its cost-effectiveness. Advancements in Head Mounted Displays (HMDs), exemplified by the recent releases of the Meta Quest 3 and Apple Vision Pro, have fueled a resurgence in VR technology. While primarily marketed for entertainment purposes, the use of Virtual Reality (VR) in drone pilot training is a promising area, with the potential to optimize learning progress [1]. However, there is a need for further research to fully explore its effectiveness, particularly in comparison to traditional training methods [2]. Despite these advancements, challenges such as VR-induced nausea during training sessions remain, potentially hindering the widespread adoption of virtual training environments.

The definition of drones can vary significantly, encompassing terrestrial, aquatic, and aerial variations, each designed for particular environments and functions. However, our study specifically focuses on aerial drones, or UAVs, distinguished by their versatility and maneuverability. With full 6 Degrees of Freedom (6-DoF) motion [3], they can move freely in three-dimensional space, allowing precise navigation, obstacle avoidance, and stable hovering. This unique capability makes aerial drones invaluable across industries, from agriculture to cinematography [4], due to their unmatched operational flexibility and efficiency in challenging environments.

Since our study focuses on professional applications of VR it is important to distinguish hobbyist and professional VR environments which differ in tasks and objectives. Hobbyist VR focuses on entertainment and leisure activities with assessment based on user experience and engagement. In contrast, professional VR is used for training, education, and business purposes, emphasizing skill development and productivity. Tasks in professional VR-training environments for UAV-Drone operations encompass various maneuvers, such as vertical takeoff and landing, hovering at specific heights for a specific duration, and precise directional control. The key distinction lies in the intended use, objectives, and formality of tasks and assessments between hobbyist and professional VR environments.

- Minh Tam Le MSc Computing Science student at the University of Groningen, E-mail: m.t.le@student.rug.nl.

- Himawan Indra Bayu MSc Computing Science student at the University of Groningen, E-mail: h.i.bayu@student.rug.nl.

1.1 Research Questions

Our study aims to address the gap in the literature concerning VR sickness in the context of drone training sessions. This gap may exist for several reasons. VR drone training is a relatively new field, and initial research might have focused primarily on its effectiveness. Additionally, existing VR sickness research might not be directly applicable due to the specific types of movements and environments encountered in drone training simulations. Furthermore, measuring VR sickness can be subjective and challenging, potentially hindering the development of a robust body of research in this area. To delve into the documented causes of VR sickness in existing literature and explore its unique impact on drone trainees, we have formulated the following research questions:

- RQ1: To what extent does VR-induced nausea affect the effectiveness of VR training for professional drone operators?
- RQ2: How do individual factors (e.g., age, prior VR experience) influence susceptibility to VR-induced nausea during drone training?
- RQ3: Are there specific drone training maneuvers or environments in VR that are more likely to induce nausea?

1.2 Review Structure

Section 2 presents related literature to provide context and essential insights before proceeding to Section 3, where our findings from the systematic literature review are outlined. Section 4 then examines how these findings answer our research questions. Finally, Section 5 offers conclusions drawn from our review paper.

2 Related Literature

In this section, we will review existing literature to provide context for our study and identify knowledge gaps. Examining prior work will inform our interpretation of the results and discussion.

2.1 VR-Induced Nausea

While today's VR technology with HMDs offers a more immersive experience, the underlying principles of VR-induced nausea, or VR sickness in literature, remain relevant. Early literature, dating back to the 1990s, laid the foundation for understanding this phenomenon, even without the advanced technology we have now. These studies, combined with the evolution of VR environments, provide a comprehensive picture of VR sickness. VR sickness is a well-documented issue that can disrupt the enjoyment of VR experiences. It manifests as nausea, dizziness, eyestrain, and disorientation [5, 7]. This discomfort arises from a mismatch between what the user sees and what their body feels. Visually, the

user may be moving through a virtual world, but physically they remain stationary. This disconnect disrupts the body's sense of balance and reduces the feeling of immersion [5]. Notably, individual susceptibility to VR sickness varies greatly [7]. Jinjakam [8] underscores the importance of thoroughly understanding these symptoms for the development of truly immersive VR environments. By incorporating the insights from early research and addressing the factors exacerbated by modern HMDs, we can create VR training environments that are both engaging and comfortable for future drone operators [11].

2.2 Drone Training Sessions

Drone training sessions in VR can accommodate diverse applications and training needs, such as inspection [12, 13], construction [14], military UAV [4] training, and more. These training sessions can offer realistic virtual spaces to ensure safe and cost-effective training while minimizing material expenses and ensuring worker safety [13]. The introduction of HMDs and VR technology has transformed traditional drone and pilot training environments, enhancing immersion beyond the traditional way in virtual environments. Figure 1 and Figure 2, illustrates VR training environments for pilots using HMD and VR.

Figure. 1. Pilot Training Environment using HMD [2]

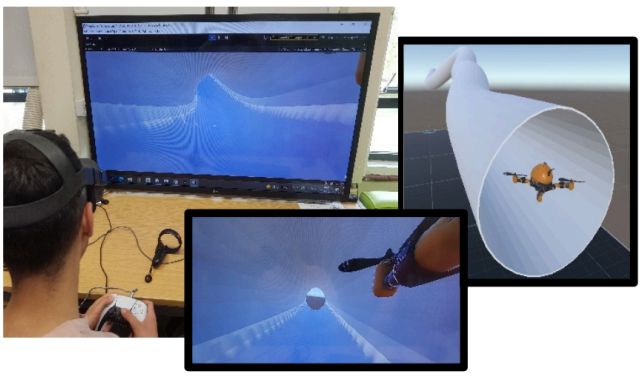


Figure 2. Drone Training Environment using HMD [2]

This new method of creating virtual training utilizes virtual reality by wearing a HMD, a gaming controller to control the movement inside the virtual environment to train future and existing pilots with essential skills for safe and effective flight training [2].

When comparing drone training in VR with other VR training applications, the primary consideration lies in the training objectives. In Drone VR training environments, the emphasis is on improving piloting skills, enhancing situational awareness, and refining decision-making in diverse flight scenarios. The environment should effectively replicate real-world conditions, enabling drone operators to practice maneuvers, emergency procedures, and tasks relevant to their specific job requirements.

2.3 VR-Induced Nausea in Drone Training Sessions

Research on VR training for drone pilots has shown promising results, with studies indicating that VR can effectively simulate real-world tasks and reduce nausea symptoms. Chaspari [9] found that VR training can lead to similar performance, mental workload, and stress levels as real-world training. Postal [10] further enhanced the user experience in UAV training tasks by developing a VR environment using Microsoft Kinect to control the drone. However, a common challenge identified in the literature is the need for further research on the optimal design elements that specifically minimize VR-induced nausea in the context of drone training simulations. While these studies suggest VR training can be effective, more research is needed to determine the most effective design features to create comfortable and long-lasting training experiences for drone pilots.

2.4 Benefits of VR Training for Drone Operators

VR training environments have successfully replicated real-life conditions, including factors like wind speed, as highlighted in study [12]. This study also developed realistic scenarios using the Unity engine for physics-based virtual training simulators, specifically for inspecting steel transmission towers. In fact, many studies [2, 12, 13, 14] commonly utilize popular game engines for developing their virtual training environments, extending their application beyond gaming development. Furthermore, Chae et al. [12] demonstrates that VR training environments overcome limitations associated with physical drone training, such as the need for expansive open areas, short drone battery life, and safety concerns related to accidents occurring when flying in close proximity to structures. Accidents during live training sessions can result in damaging the physical drone itself which would prove costly.

2.5 Drone Operations and Applications

Our literature review on drone operations revealed a significant increase in the number of professional drone pilots. This rise is likely driven by the rapidly expanding applications of drones across various sectors [15]. These applications include agriculture, delivery, construction, inspection, photography, and cinematography, public safety, including rescue operations as well as law enforcement and military operations.

3 Results

In this section, we will present our findings derived from the literature we collected, addressing how they correspond to our research questions.

3.1 Causes of VR Sickness

The VR training provides features and environments that support the users in adapting while operating a UAV-Drone that is accident-free. Besides, it enhances motor skills and muscle memory and improves the users' hand-eye coordination skills. As a result, the users can operate the drone safely, achieving satisfying results in their jobs.

There are three factors that contribute to VR sickness: content, hardware, and human indicators [27]. Content describes aspects related to the design and layout of the virtual world. Hardware factors relate to inherent measures and specifications of the used VR input- and output devices, and their perceptual impact. Human indicators, such as age, gender, and prior experience, impact the VR perception on an individual level. For the following analysis, we focus on the factor of content.

Aspects that relate to content are optical flow, graphical realism, duration, audio and task complexity. Their meaning and impact on VR perception is explained in the following subsections.

3.2 Graphic realism

Graphical realism is visualization in a virtual environment developed to give users interactive experiences. Graphic realism is constructed by shapes to create particular objects, either static or dynamic. Furthermore, Pedro et al. [30] found a correlation between the perception of details such as color, shadows, and textures and the realism of a scene in virtual reality.

In order to get a more realistic experience, some sophisticated visual scenes can be applied in virtual reality. However, it only sometimes gives a good experience for users. Complicated scenarios were employed by Kingdon et al. [23] to see users' responses. Among 1028 participants (602 males; 426 females), some vomited during or after the first exposure. Another study was conducted to observe the effects of visual realism. It was found that in a high-realism environment, which is modern graphic, the nausea score is higher than in a no-detail environment [29].

Bahit et al. [28] investigated correlations between cybersickness in day-night driving. The experiment involved ten males with an average age of 22 years old. They were asked to do a driving simulation in three different scenes: night session, morning-low, and morning-high session. Based on the SSQ score, most participants experienced nausea in the morning-high session.

3.3 Duration

Regan [18] investigated how virtual reality causes nausea and other side effects. The experiment involved 150 subjects immersed in a virtual environment for 20 minutes. The results showed that around 21% of the subjects experienced mild nausea, 5% experienced moderate nausea, and 2% experienced severe nausea.

Stanney et al. [31] investigated the effects of immersiveness in virtual environments. Among 1102 participants, 142 of the subjects dropped out because of sickness.

Min et al. [34] studied VR sickness that emerged in a graphic simulator. They did an experiment of driving a car for 60 minutes at a constant speed (60 km/h). After 10 min of the experiment began, the subjects were reported experiencing nausea.

3.4 Audio Design

Audio in a virtual environment plays a vital role in satisfying user experience when performing a task. In terms of having training with moving objects in a virtual environment such as a car, motorcycle, or drone, hearing audio from it helps to improve users' sensitivity to what they are controlling [19].

Sound effects in virtual reality facilitate a pleasant experience for users, even though it can cause a problem related to VR sickness. A study conducted by Sun et al. [20] explored the effect of hearing impairment by creating augmented speech-in-noise. Although the degree of Nausea was lower than other symptoms, this indicates that an audio issue can cause Nausea in users.

3.5 Task Complexity

Processing activities differently and moving between multiple media are an instance of multitasking [21]. An effect of Multitasking on Simulator Sickness has been studied by Açıkel et al. [22] involving sixteen students on training in an advanced aerodrome control simulator. The results show that the simulator sickness score on those who perform multitasking tasks, such as nausea, was significantly higher than the ones who do one task only.

3.6 Optical flow

Many studies have found the relationship between visual stimuli in VR environments and VR sickness, especially with nausea. Chen et al.[25] found that a high speed in a VR scene often leads to increased unpleasant feelings, even though the discomfort may decrease gradually because of the adaptation of the users.

It was also found that in virtual environments, simulated rotatory self-motion about one or two axes leads to the occurrence of VR sickness[24] or even with different speeds of objects' movement [33]

Although some debates arise concerning the correlation between unpleasantness and the sensation of illusory self-motion, some studies suggest that factors such as speed and rotational movements significantly influence VR sickness[26].

4 Discussion

The main contribution of this paper is to explore which aspects that induce nausea in the virtual environment. We explore some literature that emphasize factors that are correlated to content. By understanding the causes of nausea, we can try to reduce VR Sickness by adjusting or improving the quality of content

4.1 *To what extent does VR-induced nausea affect the effectiveness of VR training for professional drone operators?*

As a proposed solution supports professionals in adapting to virtual environments or achieving the highest score on a given task, it is necessary to bring realism onto virtual reality to provide realistic training. However, applying extreme details of objects in a virtual environment, such as the structure of a building or trees, sounds of an engine or air or sea, and providing a realistic speed can stimulate discomfort to users, such as nausea, which affects effectiveness of professionals' training.

Furthermore, a high level of task complexity also influences the effectiveness of training. By increasing the level of task, users are expected to put a lot of effort into it, which can result in arising symptoms.

Although the main purpose of training in virtual environment is supporting professional to train with null-accident environment, it is suggested that the quantity of training, especially in terms of content must be adjusted according to users

4.2 *How do individual factors (e.g., age, prior VR experience) influence susceptibility to VR-induced nausea during drone training?*

A number of measurements have been to map which human factors that influence nausea during drone training. While in some experiments women suffer nausea more than men, it remains unclear whether individual factors, such as age, gender, or VR experience has correlation with nausea because questionnaire measurement is a subjective measurement. Thus, an objective measurement is needed to ensure that human factors have correlations with emerging nausea during drone training.

4.3 *Are there specific drone training maneuvers or environments in VR that are more likely to induce nausea?*

Based on the literature review, it is assumed that training environments that have a high level of graphics induce more nausea than virtual environments with a low level of graphics. In addition, training that requires users to perform actions while following a prescribed route, such as a maze, has a higher chance of inducing nausea in users. Furthermore, controlling drones with high speed also has an impact on users' comfort.

5 Conclusion

In conclusion, addressing VR-induced nausea is crucial for ensuring the effectiveness and success of VR training sessions for professional UAV-Drone operators. This review focused on the content factor and the impact of its aspects on VR-induced nausea. For a concise design guide on UAV-VR training environments, the other factors of human indicators and hardware also need to be considered.

This literature review highlights the importance of realistic training environments while also considering the potential negative effects of extreme details and high task complexity on inducing nausea. It is essential to strike a balance between realism and user comfort to optimize training effectiveness.

Moreover, individual factors like age, gender, and prior VR experience may influence susceptibility to VR-induced nausea, emphasizing the need for personalized training approaches. Objective measurements are necessary to determine the correlation between these factors and the onset of nausea during drone training accurately.

By identifying specific drone training maneuvers or environments that are more likely to induce nausea, training programs can be tailored to minimize discomfort and maximize learning outcomes. Strategies to reduce VR-induced nausea, such as adjusting content quality and quantity, optimizing graphics, and controlling speed, can significantly enhance the training experience for UAV-Drone operators.

Overall, by addressing VR-induced nausea through targeted interventions and personalized approaches, the effectiveness and efficiency of VR training for professional drone operators can be greatly improved, leading to safer and more successful operations in various sectors.

References

[1] E. v. Weelden, M. Alimardani, T. J. Wiltshire and M. M. Louwerse, "Advancing the Adoption of Virtual Reality and Neurotechnology to Improve Flight Training," 2021 IEEE 2nd International Conference on Human-Machine Systems (ICHMS), Magdeburg, Germany, 2021, pp. 1-4, doi: 10.1109/ICHMS53169.2021.9582658.

[2] S. Livatino et al., "Immersive Visualization in Pilot Training: from Cockpit Panels to Drone Navigation" 2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE), Rome, Italy, 2022, pp. 454-458, doi: 10.1109/MetroXRAINE54828.2022.9967530.

[3] X. Xia et al., "A 6-DOF Telexistence Drone Controlled by a Head Mounted Display," 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 2019, pp. 1241-1242, doi: 10.1109/VR.2019.8797791.

[4] Javaid, Mohd & Haleem Khan, Ibrahim & Singh, Ravi & Rab, Shanay & Suman, Rajiv. (2021). Exploring contributions of drones towards Industry 4.0. Industrial Robot: the international journal of robotics research and application. ahead-of-print. 10.1108/IR-09-2021-0203.

[5] Kolasinski, Eugenia M.. "Simulator Sickness in Virtual Environments." (1995).

[6] Lawrence J. Hettinger, Gary E. Riccio; Visually Induced Motion Sickness in Virtual Environments. Presence: Teleoperators and Virtual Environments 1992; 1 (3): 306–310. doi: https://doi-org.proxy-ub.rug.nl/10.1162/pres.1992.1.3.306

[7] Lampton, Donald R. et al. "Side Effects and Aftereffects of Immersion in Virtual Environments." Proceedings of the Human Factors and Ergonomics Society Annual Meeting 38 (1994): 1154 - 1157.

[8] Jinjakam, Chompoonuch and Kazuhiko Hamamoto. "Simulator sickness in immersive virtual environment." The 5th 2012 Biomedical Engineering International Conference (2012): 1-4.

[9] Sakib, Md. Nazmus et al. "An Experimental Study of Wearable Technology and Immersive Virtual Reality for Drone Operator Training." (2020).

[10] Postal, Guilherme Riter et al. "A Virtual Environment for Drone Pilot Training Using VR Devices." 2016 XVIII Symposium on Virtual and Augmented Reality (SVR) (2016): 183-187.

[11] Smolyanskiy, N., & González-Franco, M. (2017). Stereoscopic first person view system for drone navigation. Frontiers in Robotics and AI, 4. https://doi.org/10.3389/frobt.2017.00011 .

[12] Chae, CH., Ko, KH. Development of Physics-Based Virtual Training Simulator for Inspections of Steel Transmission Towers. J. Electr. Eng. Technol. (2023). https://doi-org.proxy-ub.rug.nl/10.1007/s42835-023-01692-9

[13] Y. Li, M. M. Karim and R. Qin, "A Virtual-Reality-Based Training and Assessment System for Bridge Inspectors With an Assistant Drone," in IEEE Transactions on Human-Machine Systems, vol. 52, no. 4, pp. 591-601, Aug. 2022, doi: 10.1109/THMS.2022.3155373.

[14] Onososen, A., Musonda, I., Ramabodu, M., Dzuwa, C. (2023). Safety and Training Implications of Human-Drone Interaction in Industrialised Construction Sites. In: Skatulla, S., Beushausen, H. (eds) Advances in Information Technology in Civil and Building Engineering. ICCCBE 2022. Lecture Notes in Civil Engineering, vol 358. Springer, Cham. https://doi-org.proxy-ub.rug.nl/10.1007/978-3-031-32515-1_20

[15] Custers, Bart. (2016). The Future of Drone Use: Opportunities and Threats from Ethical and Legal Perspectives. 10.1007/978-94-6265-132-6.

[16] Caserman, P., Martinussen, M., Göbel, S. (2019). Effects of End-to-end Latency on User Experience and Performance in Immersive Virtual Reality Applications. In: van der Spek, E., Göbel, S., Do, EL., Clua, E., Baalsrud Hauge, J. (eds) Entertainment Computing and Serious Games. ICEC-JCSG 2019. Lecture Notes in Computer Science(), vol 11863. Springer, Cham. https://doi-org.proxy-ub.rug.nl/10.1007/978-3-030-34644-7_5

[17] Davis S, Nesbitt K, Nalivaiko E (2015) Comparing the onset of cybersickness using the oculus rift and two virtual roller coasters. Proceedings of the 11th Australasian conference on interactive entertainment 27

[18] Regan, C. An investigation into nausea and other side-effects of head-coupled immersive virtual reality. Virtual Reality 1, 17–31 (1995). https://doi-org.proxy-ub.rug.nl/10.1007/BF02009710

[19] Davis, E. T., Scott, K., Pair, J., Hodges, L. F., & Oliverio, J. (1999). Can Audio Enhance Visual Perception and Performance in a Virtual Environment? Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 43(22), 1197-1201. https://doi-org.proxy-ub.rug.nl/10.1177/154193129904302206

[20] Katja Rogers, Giovanni Ribeiro, Rina R. Wehbe, Michael Weber, and Lennart E. Nacke. 2018. Vanishing Importance: Studying Immersive Effects of Game Audio Perception on Player Experiences in Virtual Reality. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18). Association for Computing Machinery, New York, NY, USA, Paper 328, 1–13. https://doi-org.proxy-ub.rug.nl/10.1145/3173574.3173902

[21] Ophir E, Nass C, Wagner AD. Cognitive control in media multitaskers. Proc Natl Acad Sci U S A. 2009 Sep 15;106(37):15583-7. doi: 10.1073/pnas.0903620106. Epub 2009 Aug 24. PMID: 19706386; PMCID: PMC2747164.

[22] Yörük Açıkel, B., Turhan, U., & Akbulut, Y. (2018). Effect of Multitasking on Simulator Sickness and Performance in 3D Aerodrome Control Training. Simulation & Gaming, 49(1), 27-49. https://doi-org.proxy-ub.rug.nl/10.1177/1046878117750417

[23] Kingdon, K. S., Stanney, K. M., & Kennedy, R. S. (2001). Extreme Responses to Virtual Environment Exposure. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 45(27), 1906-1910. https://doi-org.proxy-ub.rug.nl/10.1177/154193120104502711

[24] Bonato, F., Bubka, A., & Palmisano, S. (2009). Combined Pitch and Roll and Cybersickness in a Virtual Environment. Aviation, Space, and Environmental Medicine, 80(11), 941-945. https://doi.org/10.3357/asem.2394.2009

[25] Chen, W. & Chen, J.Z. & So, Richard. (2011). Visually induced motion sickness: Effects of translational visual motion along different axes. Contemporary Ergonomics and Human Factors 2011. 281-287. 10.1201/b11337-47.

[26] Akizuki, H., Uno, A., Arai, K., Morioka, S., Ohyama, S., Nishiike, S., Tamura, K., & Takeda, N. (2005). Effects of immersion in virtual reality on postural control. Neuroscience Letters, 379(1), 23-26. https://doi.org/10.1016/j.neulet.2004.12.041
[27] Chang, E., Kim, H. T., & Yoo, B. (2020). Virtual Reality Sickness: A Review of Causes and Measurements. International Journal of Human–Computer Interaction, 36(17), 1658-1682. https://doi.org/10.1080/10447318.2020.1778351
[28] M. Bahit, S. Wibirama, H. A. Nugroho, T. Wijayanto and M. N. Winadi, "Investigation of visual attention in day-night driving simulator during cybersickness occurrence," 2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, Indonesia, 2016, pp. 1-4, doi: 10.1109/ICITEED.2016.7863260.
[29] M. Pouke, A. Tiiro, S. M. LaValle and T. Ojala, "Effects of Visual Realism and Moving Detail on Cybersickness," 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Tuebingen/Reutlingen, Germany, 2018, pp. 665-666, doi: 10.1109/VR.2018.8446078.
[30] Pardo, P. J., Suero, M. I., & Pérez, Ángel L. (2018). Correlation between perception of color, shadows, and surface textures and the realism of a scene in virtual reality. Journal of the Optical Society of America A, 35(4), B130. https://doi.org/10.1364/josaa.35.00b130
[31] Stanney, K. M., Hale, K. S., Nahmens, I., & Kennedy, R. S. (2003). What to expect from immersive virtual environment exposure: influences of gender, body mass index, and past experience. Human factors, 45(3), 504–520. https://doi.org/10.1518/hfes.45.3.504.27254
[32] Sun, K., Pontoppidan, N. H., Wendt, D., & Bramsløw, L. (2022). Perception of virtual reality based audiovisual paradigm for people with hearing impairment. BNAM, 1, 95-104.
[33] So, R. H. Y., Lo, W. T., & Ho, A. T. K. (2001). Effects of Navigation Speed on Motion Sickness Caused by an Immersive Virtual Environment. Human Factors, 43(3), 452-461. https://doi.org/10.1518/001872001775898223
[34] Min, B. C., Chung, S. C., Min, Y. K., & Sakamoto, K. (2004). Psychophysiological evaluation of simulator sickness evoked by a graphic simulator. Applied ergonomics, 35(6), 549–556. https://doi.org/10.1016/j.apergo.2004.06.002

Max-trees and mixture models

Paul D. Teeninga

Abstract— We explore if a max-tree, in combination with the Expectation-Maximization algorithm (EM-algorithm), can be used to deblend stars and galaxies (objects) in sky surveys. Images are modeled as mixtures of 2D Gaussian functions plus noise, and the goal is to extract each function. Each statistically significant peak in the image, a connected component at some threshold, is assumed to be a separate object. The number of significant peaks depends on noise, and how close peaks are to each other. Statistics of connected components, at different threshold levels, can be efficiently computed using a max-tree. After computing initial estimates of parameters in the Gaussian mixture, the EM-algorithm for Gaussian mixtures improves estimates. To reduce effects of noise, pixels that are relatively far from objects are excluded using the Mahalanobis distance. When testing the method on images with dense 2D Gaussian functions plus noise, if every object is detected, objects are modeled correctly in most cases, even if the error is in the 90% quantile. If not all objects are detected, objects that are detected are modeled plausibly. When applied to a real image from a sky survey, many objects are not correctly modeled by a 2D Gaussian function, and more work is needed. Alternatively, mixture models can be used as probability distributions to extract individual objects from images without modeling.

Index Terms—Mathematical morphology, component trees, attribute filters, astronomical imaging.

1 Introduction

In astronomy, sky surveys produce a large amount of data, and there is demand for automated detection and cataloging of stars and galaxies (objects) [1] [4]. Objects can have considerable overlap (Fig. 1). To accurately compute properties of individual objects, objects are deblended, which requires shape assumptions. More specifically, grayscale images are modeled as the sum of 2D functions, one for each object, plus noise. A 2D function can, for example, be a Gaussian function. In current methods [8] [13], objects are detected and segmented with SExtractor [3] or MTObjects [15], and initial estimates of 2D functions are improved with least-squares optimization in Galfit [12]. A comparison of source-extracting tools is given in [6]. In this paper we explore if MTObjects can be used in combination with the Expectation-Maximization algorithm (EM-algorithm) [7] to deblend objects.

Fig. 1: Zoomed in view of the Perseus cluster of galaxies. Photo: ©ESA.

MTObjects uses the max-tree data structure [14]. To briefly explain what a max-tree is, connected components (or simply components) in an image are connected pixels with weights that are above some threshold. Stars and galaxies can be represented by components. A *max-tree* is a hierarchical representation of components in all threshold sets of a grayscale image (Fig. 2), and is used to efficiently compute properties, or *attributes*, of components. The name 'MTObjects' encapsulates attribute computation, detection and segmentation of objects in a max-tree. To find objects, each branch in the max-tree that is unlikely to be noise, is assumed to be a separate object.

In this paper, we model objects as 2D Gaussian functions. While this is not accurate in sky surveys, potential of the method (MTObjects plus EM-algorithm) can still be shown. In a later section we discuss how different models can be used. Initial parameter estimates of models are calculated using a max-tree. For each cluster of objects (compound object), estimates are optimized using the EM-algorithm for Gaussian mixtures. To reduce negative effects of noise on estimates, pixels that are relatively far away from objects are excluded, using the Mahalanobis distance. The output is a collection of mixture models, one for each compound object.

The rest of the paper is organized as follows. First, definitions are recalled in the preliminaries section (Section 2). Second, an implementation of the method is given (Section 3). Third, we describe the data set, test parameters in the method, and show results (Section 4). Fourth, we discuss results and future work (Section 5). Finally, we give conclusions (Section 6).

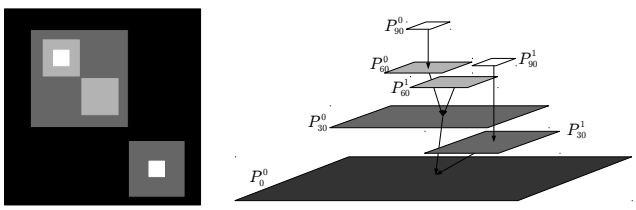


Fig. 2: Grayscale input image (left) and max-tree representation (right). Each P^i_t is a pixel set representing connected component i at threshold t. From [5].

2 Preliminaries

We recall definitions. The grayscale image and max-tree definition are from [18], and attribute definitions are based on definitions in [9].

2.1 Image

A *grayscale image* G with k dimensions is a connected node-weighted undirected graph (V,E,w), where $V \subset \mathbb{N}^k$ is the set of *nodes* (pixel coordinates), $E \subseteq V \times V$ is the set of *edges* (neighbor relations), and $w : V \to \mathbb{R}_{\geq 0}$ is a function mapping nodes to gray values. Edges in undirected graphs are symmetrically closed: $(x,y) \in E \equiv (y,x) \in E$.

• *Paul Teeninga is a MSc Computing Science student at the University of Groningen. E-mail: p.d.teeninga@gmail.com.*

An *E-path* is a sequence of nodes, where each consecutive pair is an element of E. Two distinct nodes are *connected* if an E-path exists between the nodes. A *component* is a maximal node subset where each distinct node pair is connected. Graph G is connected if and only if G contains exactly one component. When thresholding G at threshold t, the result is a subgraph $G_t = (V_t, E_t)$, where $V_t = \{x \in V \mid w(x) \geq t\}$, and $E_t = E \cap V_t \times V_t$.

2.2 Max-tree

Let $G = (V, E, w)$ be a grayscale image. We assume that w maps to distinct values (w is injective). If $w(x)$ is not injective, a new injective weight function $w'(x) = (w(x), x)$ can be constructed where weights are lexically ordered. In other sections we assume that $w(x)$ maps to nonnegative real numbers. Let W be the set of possible weights.

Definition 1 (component root) *For any component $C \subseteq V_t$, at any threshold $t \in W$,* $\text{root}(C) = \arg\min\{w(x) \mid x \in C\}$.

Every component C is associated with a node x, where $\text{root}(C) = x$ and $\text{root}^{-1}(x) = C$.

Definition 2 (parent candidate path) *A parent candidate path between nodes x and y is an E-path $(x, \ldots, y)$, where $w(x) > w(y)$, and all other nodes in the path have weights greater than $w(x)$.*

Node y at the end of a parent candidate path represents a component C' at a lower threshold. Component C' is a superset of $C = \text{root}^{-1}(x)$.

Definition 3 (parent candidates)

$$\text{cands}_G(x) = \{y \in V \mid \textit{exists a parent candidate path between } x \textit{ and } y\}$$

Parent candidates surround the component given by $C = \text{root}^{-1}(x)$ in G, and the candidate with maximum weight represents the parent component.

Definition 4 (parent)

$$\text{parent}_G(x) = \begin{cases} \arg\max\{w(y) \mid y \in \text{cands}_G(X)\}, & \textit{if } \text{cands}_G(x) \neq \emptyset \\ x, & \textit{otherwise} \end{cases}$$

Definition 5 (max-tree)

$$\text{maxtree}(G) = (V, \{(x, y = \text{parent}_G(x)) \mid x \in V \wedge x \neq y\})$$

The node with minimum weight is the root of the max-tree, and the parent of the root is the root itself, to ensure that the parent is defined for every node. Lemma 1 is a useful result showing the equivalency between subtrees and components. Proof is given in [18].

Lemma 1 *Let the root path of any node $x \in V$ be the sequence $(x, \text{parent}_G(x), \ldots)$ with no repetitions, and let C be any component in G at any threshold. The statement* $\text{root}(C)$ *is in the root path of x is equivalent to $x \in C$.*

Sequential algorithms to construct a max-tree are given in [2] [16] [20], and parallel algorithms in [5] [18]. Typically, each pixel in a 2D image has 4 or 8 neighbors. In this case, sequential algorithms run in $O(n \log n)$ time, where n is the number of pixels. Other types of connectivity are described in [11].

2.3 Max-tree attributes

Let $z : V \to Z$ be an attribute function for nodes in a max-tree $T = (V, P)$, and let $\oplus : Z \times Z \to Z$ be a binary associative function. For any pair of nodes (x, y), y is an *ancestor* of x, and x is a *descendant* of y, if a P-path exists between x and y. An ancestor or descendant is *direct* if the P-path has length two.

Definition 6 (rootfix attribute) *Let $x_0, x_1, \ldots$ be all ancestors of node $x \in V$, ordered by distance to the root. The result of $z(x_0) \oplus z(x_1) \oplus \ldots \oplus z(x)$ is a rootfix attribute of x.*

Definition 7 (leaffix attribute) *A leaffix attribute of node $x \in V$ is recursively defined as the $\oplus$ operation applied to $z(x)$ and an arbitrary order of leaffix attributes of direct descendants of x. Any leaf node y has attribute $z(y)$.*

To give a simple example, the *area* attribute of a node can be computed as leaffix attribute, where $z(x) = 1$ and $\oplus = +$. The area is the number of pixels in component $C = \text{root}^{-1}(y)$. Because an arbitrary order is used in leaffix attributes to evaluate branches, results may not be unique if $\oplus$ is not commutative.

Using the injective weight function $w(x)$ of the original image, rootfix attributes can be efficiently computed by iterating over nodes in ascending order by $w(x)$, and leaffix attributes in descending order. Parallel algorithms are given in [17].

2.4 Gaussian mixture

Let $\mathcal{N}$ be a k-variate normal distribution with parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \Sigma)$, where $\boldsymbol{\mu} \in \mathbb{R}^k$ and $\Sigma \in \mathbb{R}^{k \times k}$ is a positive semidefinite matrix. The *Mahalanobis distance* of point $\boldsymbol{x} = (x_0, \ldots, x_{k-1})^{\mathrm{T}} \in \mathbb{R}^k$ is

$$\text{mdist}(\boldsymbol{x}; \boldsymbol{\theta}) = \sqrt{(\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}} \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu})}.$$

The probability density function (pdf) of $\mathcal{N}$ is

$$\text{mvnpdf}(\boldsymbol{x}; \boldsymbol{\theta}) = (2\pi)^{-k/2} \det(\Sigma)^{-1/2} \exp\left(-\frac{1}{2} \text{mdist}(\boldsymbol{x}; \boldsymbol{\theta})^2\right).$$

We use 'Gaussian function' to refer to mvnpdf multiplied by a scale factor. The *Gaussian mixture pdf* with parameters $\boldsymbol{\theta}' = (\boldsymbol{\beta}, \boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_{m-1})$ is

$$\text{gmpdf}(\boldsymbol{x}; \boldsymbol{\theta}') = \sum_{j=0}^{m-1} \beta_j \text{mvnpdf}(\boldsymbol{x}; \boldsymbol{\theta}_j), \tag{1}$$

a linear combination of multivariate normal pdfs, where β coefficients are positive and sum to one. Let $X = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}) \in \mathbb{R}^{k \times n}$ be a random sample from a gaussian mixture distribution. The likelihood of Gaussian mixture parameter estimates $\hat{\boldsymbol{\theta}}$ given X is

$$\mathcal{L}(\hat{\boldsymbol{\theta}}; X) = \prod_{i=0}^{n-1} \text{gmpdf}(\boldsymbol{x}_i; \hat{\boldsymbol{\theta}}). \tag{2}$$

3 Implementation

We begin by finding statistically significant components in an image, based on the method in [15]. Next, compound objects are segmented and initial estimates of object parameters computed. Finally, for each compound object, parameter estimates are optimized. The output is a collection of compound objects described by mixture models.

3.1 MTObjects

The data structure of a max-tree is an array of parent references. Each node in the max-tree has a number in $[0..\#V)$, and a parent reference is a number. Image weights are assumed to be the sum of nonnegative 2D functions plus normally distributed noise. Independently per pixel, noise has zero mean and variance $\sigma^2(x)$. The variance is dependent on the sum of functions at x. The *power* attribute of node $x \in V$ is defined as

$$\text{power}(x) = \sum_{y \in \text{root}^{-1}(x)} (w(y) - w(\text{parent}(x)))^2,$$

where $\text{root}^{-1}(x)$ is the component with root x. To compute the power attribute as leaffix attribute, let $\Delta(x) = w(x) - w(\text{parent}(x))$. Then, $z(x) = (x, 1, \Delta(x), \Delta(x)^2)$ and Alg. 1 is the $\oplus$ operation. The resulting attribute tuple for node x contains respectively x, and the area, volume, and power attributes.

To find components that are unlikely to be only noise, suppose the bivariate function in the area of a component has the same value as

Algorithm 1: Merging power attributes

Input: *lhs* and *rhs*, elements of Z
Output: evaluation of $lhs \oplus rhs$

▷ swap *lhs* and *rhs* if $w(rhs.x) < w(lhs.x)$
$(to, area_0, vol_0, pow_0) \leftarrow lhs$
$(from, area_1, vol_1, pow_1) \leftarrow rhs$
$\delta \leftarrow w(\text{parent}(from)) - w(\text{parent}(to))$ ▷ Nonnegative
$area_r = area_0 + area_1$
$vol_r = vol_0 + vol_1 + \delta \cdot area_1$
$pow_r = pow_0 + pow_1 + 2 \cdot \delta \cdot vol_1 + \delta^2 \cdot area_1$
return $(\text{to}, \text{area}_r, \text{vol}_r, \text{pow}_r)$

the parent component. In this case, the power attribute divided by the noise variance has a χ^2 distribution, and the component is rejected as noise if the power attribute is much larger than expected. For example, if $\text{chi2inv}(1-10^{-6}, \text{area}(x)) < \text{power}(x)/\sigma^2(\text{parent}(x))$, where chi2inv is the inverse of the χ^2 cumulative distributive function, and the degrees of freedom are equal to $\text{area}(x)$. We refer to rejected nodes as significant nodes.

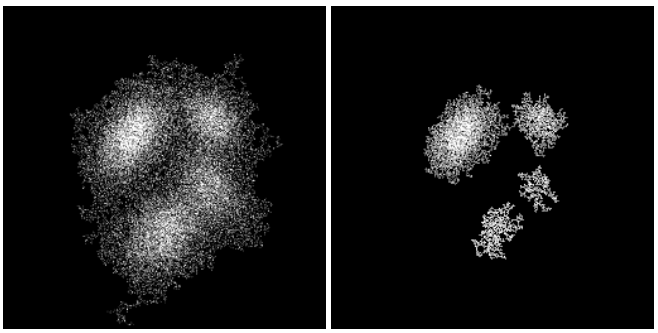


Fig. 3: Segmented compound object (left) and object cores (right).

3.2 Segmentation and initial parameters

For each compound object, which is represented by a node in the max-tree, an array of coordinates is extracted. In a copy of the parents array, *P*-paths are contracted to the last significant ancestor (excluding the root), representing a compound object. This is implemented as a rootfix attribute. After path contraction, nodes are iterated over and coordinates are copied for each compound object. Fig. 3 (left) shows an extracted compound object.

Object cores are found by selecting components closest to the root that are significant, and additionally have the condition that there is exactly one branch with significant nodes (Fig. 3, right). Because there is no branching to multiple objects, there is no overlap. For more details see the source code, which is available upon request.

Each object core is assumed to correspond to a scaled bivariate normal probability density function, a 2D Gaussian function. However, it is possible that two 'object cores' are in fact parts of the same function, because of noise. This can happen when an object has a small diameter, and noise cuts the object (Fig. 9). Alg. 2 shows how parameter estimates are calculated. These estimates are used as input for parameter optimization. Mean coordinates are also computed as leaffix attribute.

3.3 Parameter optimization

The EM-algorithm iteratively increases the likelihood function of a Gaussian mixture $\mathscr{L}(\hat{\boldsymbol{\theta}}; X = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{m-1}))$ (Eq. 2) by improving parameter estimates $\hat{\boldsymbol{\theta}}$ [7]. In the E-step, per data point $\boldsymbol{x}_i$, and for each normal distribution $\mathscr{N}_j$, the probability is computed that $\boldsymbol{x}_i$ was generated by $\mathscr{N}_j$, using current parameter estimates. In the M-step, parameter estimates are recalculated using the probabilities in the E-step.

Algorithm 2: Initial mixture model parameter estimates

Input: Node attributes of m object cores: area, volume and mean coordinates
Output: Parameter estimates s (scale) and $\hat{\boldsymbol{\theta}}$

for $j \in [0..m)$ **do**
  $x \leftarrow$ max-tree node corresponding to object core j
  $\hat{\boldsymbol{\mu}}_j \leftarrow$ mean coordinates of component $C = \text{root}^{-1}(x)$
  ▷ Constant c determined in results
  $S_j \leftarrow c \cdot \text{area}(x) I$ ▷ Identity matrix I
  $\hat{\beta}_j \leftarrow \text{volume}(x)$
end
$s \leftarrow$ sum of $\hat{\boldsymbol{\beta}}$
$\hat{\boldsymbol{\beta}} \leftarrow \hat{\boldsymbol{\beta}}/s$
$\hat{\boldsymbol{\theta}} \leftarrow (\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\mu}}_0, S_0, \ldots, \hat{\boldsymbol{\mu}}_{m-1}, S_{m-1})$
return s and $\hat{\boldsymbol{\theta}}$

In the context of a 2D image, a pixel with coordinate $x \in V$ contains data points in a 1×1 window, and weight $w(x)$ is the fractional number of data points. If an image weight is zero, the pixel is not used in calculations. Alg. 3 shows the implementation based on methods in [7]. Data points are approximated by the coordinate of the pixel. Conceptually, in an iteration, each bivariate function is extracted from the image using current parameter estimates, and parameters are recalculated using the extracted bivariate functions. Whenever a parameter estimate is invalid, for example when there is a division by zero, the object is removed by changing the β coefficient to zero. Scale s is an estimate of the sum of image weights without noise.

We explain the purpose of using Mahalanobis distance to exclude pixels. Since the weight function is nonnegative, noise further away from objects can result in inaccurate parameter estimates. Therefore, using Mahalanobis distance, we only use pixels close to objects in an elliptical window. Pixel weights outside the window are assumed to be identical to the model. The density and variance fractions of data points outside of Mahalanobis distance d are determined analytically using polar coordinates. It can be shown that the density fraction outside of d is $\exp(-d^2/2)$, and the variance fraction outside of d is $(d^2/2+1)\exp(-d^2/2)$. Using the density and variance fractions, parameter estimates are adjusted at the end in Alg. 3 (correction for excluded data). Since the Mahalanobis distance is calculated with current estimates, the elliptical window changes in each iteration.

Because the coordinate set changes in each iteration, the likelihood function is not used to stop the algorithm. Instead, the algorithm stops when the distance between consecutive parameter estimates is a below a threshold, which can be implemented in different ways. We verify experimentally in the results section that the algorithm still converges, and that parameter estimates are improved. If the maximum number of iterations is fixed, Alg. 3 applied to all compound objects in an image runs in $O(nm)$ time, where n is the number of pixels, and m is the maximum number of objects in a compound object. The combination with max-tree attributes runs in $O(nm + n\log n)$ time.

4 Results

We first describe test images and the error measure. Second, we test how changing Mahalanobis distance d impacts the error, and if there is a difference between underestimating the covariance matrix compared to overestimating. This is accomplished without MTObjects, since the actual number of objects and parameters are known. Third, we find a good value for the area fraction in covariance matrix estimates in MTObjects. Fourth, we test if 2D Gaussian functions in noisy images can be deblended. Finally, we apply the method to a real astronomy image.

Algorithm 3: Deblend 2D Gaussian functions in noisy images

Input: $2 \times n$ coordinate matrix $X = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1})$, nonnegative weight function $w(\boldsymbol{x})$, Mahalanobis distance d, scale estimate s^0, parameter estimates $\hat{\boldsymbol{\theta}}^0 = (\hat{\boldsymbol{\beta}}^0, \hat{\boldsymbol{\mu}}_0^0, S_0^0, \ldots, \hat{\boldsymbol{\mu}}_{m-1}^0, S_{m-1}^0)$.

Output: Optimized estimates

▷ `Fractions outside Mahalanobis distance` d

$densFraction \leftarrow \exp(-d^2/2)$

$varFraction \leftarrow densFraction \cdot (d^2/2+1)$

$k \leftarrow 0$

do

  for $j \in [0..m)$ **do**

    $\hat{\boldsymbol{\mu}}_j^{k+1} \leftarrow 0$

    $S_j^{k+1} \leftarrow 0$

    $flux[j] \leftarrow 0$

  end

  for $i \in [0..n)$ **do**

    if $w(\boldsymbol{x}_i) = 0$ **then**

      continue next i

    end

    $compound \leftarrow 0$

    for $j \in [0..m)$ **do**

      $compound$ += $\hat{\beta}_j^k \text{mvnpdf}(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}_j^k, S_j^k)$

    end

    for $j \in [0..m)$ **do**

      if $\text{mdist}(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}_j^k, S_j^k) > d$ **then**

        continue next j

      end

      $proto \leftarrow w(\boldsymbol{x}_i)\hat{\beta}_j^k \text{mvnpdf}(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}_j^k, S_j^k)$ / $compound$

      $\boldsymbol{c} \leftarrow \boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j^k$

      $flux[j]$ += $proto$

      $\hat{\boldsymbol{\mu}}_j^{k+1}$ += $proto \cdot \boldsymbol{x}_i$

      $S_j^{k+1}[0,0]$ += $proto \cdot c_0 c_0$

      $S_j^{k+1}[0,1]$ += $proto \cdot c_0 c_1$

      $S_j^{k+1}[1,0]$ += $proto \cdot c_1 c_0$

      $S_j^{k+1}[1,1]$ += $proto \cdot c_1 c_1$

    end

  end

  $s^{k+1} \leftarrow$ sum of elements in $flux$

  ▷ `Rescale and correct for excluded data`

  for $j \in [0..m)$ **do**

    $\hat{\beta}_j^{k+1} \leftarrow flux[j]/s^{k+1}$

    $\hat{\boldsymbol{\mu}}_j^{k+1} \leftarrow \hat{\boldsymbol{\mu}}_j^{k+1}/flux[j]$

    $S_j^{k+1} \leftarrow (1 - densFract)/(1 - varFract) S_j^{k+1}/flux[j]$

  end

  $s^{k+1} \leftarrow s^{k+1}/(1 - densFract)$

  k += 1

while *distance between* $(s_k, \hat{\boldsymbol{\theta}}^k)$ *and* $(s^{k-1}, \hat{\boldsymbol{\theta}}^{k-1}) > \varepsilon$;

return s^k, $\hat{\boldsymbol{\theta}}^k$

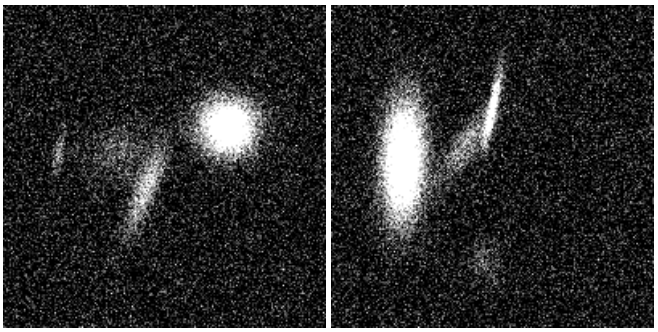


Fig. 4: Two examples of test images with four 2D Gaussian functions plus noise. Showing weights in $[0,4]$.

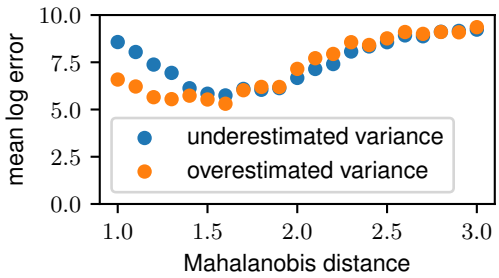


Fig. 5: Mean log error versus Mahalanobis distance, after parameter optimization. Initial covariance matrices are underestimated or overestimated by a factor of 4. The error is the sum of squares of weight differences between the ground truth and mixture model. Each dot is the mean of 200 log errors.

4.1 Test images and error measure

Test images consist of four random 2D Gaussian functions plus normally distributed noise with variance 1. Negative weights are changed to zero, consistent with segmentations by MTObjects. Image dimensions are 201×201. Variances in the directions of covariance eigenvectors are between 1 and 25^2. The mean is chosen such that the image contains most of the function. Functions have a weight between 1 and 10 at the mean. Fig. 4 shows two examples. The data set is intended to model a subset of objects in sky surveys, although the parameter distribution of those objects is likely different.

Let w' be the weight function of a ground truth image without noise. The error is calculated using weight differences of the ground truth and mixture model evaluated at every pixel:

$$\sum_{i=0}^{n-1} (w'(\boldsymbol{x}_i) - s \cdot \text{gmpdf}(\boldsymbol{x}_i; \hat{\boldsymbol{\theta}}))^2,$$

where $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}$ are pixel coordinates, s and $\hat{\boldsymbol{\theta}}$ are estimates from Alg. 3, and gmpdf is the Gaussian mixture pdf (Eq. 1).

4.2 Mahalanobis distance

To determine a good value for Mahalanobis distance d in Alg. 3, initial parameter estimates are equal to the actual parameters, except for covariance matrices, which are either underestimated or overestimated by a factor of 4. Alg. 3 optimizes parameter estimates, and the resulting mean log error versus d is shown in Fig.5. Each dot is the mean of 200 log errors. The algorithm stops when the Euclidean distance between consecutive parameter estimates is below 10^{-3}, or after 50 iterations. We observed convergence in every case, however we limit the number of iterations to obtain larger samples. Overestimating the covariance matrix results in less error when d is below 1.7. When either underestimating or overestimating the covariance matrix, $d = 1.6$

seems to be nearly optimal, and this value is used in the following experiments. The density (or flux) fraction within this distance is approximately 0.72.

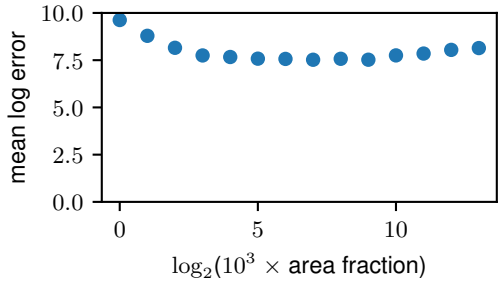


Fig. 6: Mean log error versus area fraction, after parameter optimization. The diagonal values in the initial covariance matrix are initialized with a fraction of the area attribute in MTObjects. Mahalanobis distance 1.6. The error is the sum of squares of weight differences between the ground truth and mixture model. Each dot is the mean of 200 log errors.

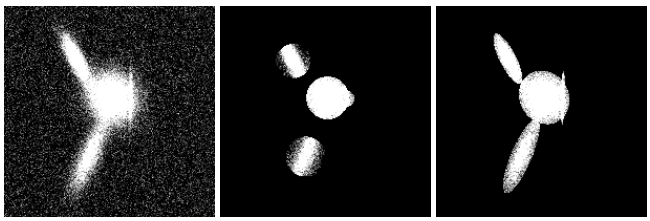


Fig. 7: Example with a relatively low log error of 6.4. Input image (left), pixels within Mahalanobis distance 1.6 using initial parameter estimates (middle), and final estimates (right). Area fraction $c = 0.25$.

4.3 Covariance matrix estimate

The area attribute in MTObjects is used to give an initial approximation of the covariance matrix. The approximation is not accurate, and is only intended as starting point for Alg. 3. If x is the max-tree node representing an object core, then the covariance matrix is initialized with $c \cdot \text{area}(x)I$, where c is the area fraction, and I is the identity matrix. Fig. 6 shows the mean log error versus area fraction. The error is much larger compared to Fig.5, since MTObjects does not always detect every object. The error decreases until an area fraction of approximately 0.1, and increases again after an area fraction of approximately 0.5. Area fraction $c = 0.25$ is used in the following experiments, a value in the middle, unless mentioned otherwise. We assume that the slight error increase at this value in the graph is due to variance. Fig. 7 shows an example with a relatively low log error.

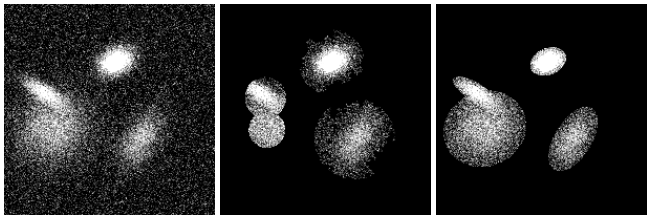


Fig. 8: Example when four 2D Gaussian functions are detected with a relatively high log error of 7.2. Input image (left), pixels within Mahalanobis distance 1.6 using initial parameter estimates (middle), and final estimates (right). Area fraction $c = 0.25$.

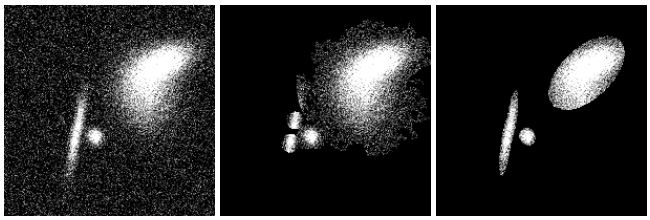


Fig. 9: Rare case when a 2D Gaussian function has two separate peaks. Input image (left), pixels within Mahalanobis distance 1.6 using initial parameter estimates (middle), and final estimates (right). Area fraction $c = 0.25$.

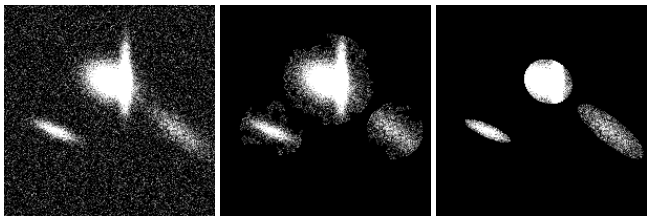


Fig. 10: Example when a 2D Gaussian function is not detected separately. Log error 9.3. Input image (left), pixels within Mahalanobis distance 1.6 using initial parameter estimates (middle), and final estimates (right). Area fraction $c = 0.25$.

4.4 Extracted mixture models

Bivariate Gaussian functions are referred to as objects. In a sample of 1000 images, MTObjects detected one to five objects in respectively 17, 226, 464, 290, and 3 images. Respective 90% quantile log errors, when one to four objects are detected, are 10.1, 9.3, 8.7, and 7.2. Clearly, when images contain heavily blended objects, MTObjects does not detect the correct amount of objects in most cases. However, objects that are detected can still be modeled. When four objects are detected, extracted mixture models remain plausible when visually inspecting results with a log error in the 90% quantile. Fig. 8 shows an example. Fig. 9 shows a rare case where two peaks are detected in one object. Fig. 10 shows a result when only three objects are detected with a log error in the 90% quantile. Models for the three detected objects are plausible.

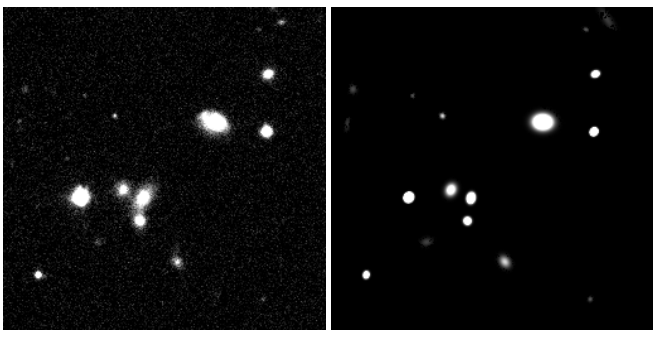


Fig. 11: Real input image (left) and mixture model (right). Area fraction $c = 0.25$. File `fpC-001231-r1-0109.fit` from [1].

4.5 Real image

Fig. 11 shows a real example from the Sloan Digital Sky Survey [1]. There are two problems in the result. First, objects are not correctly modeled. A 2D Gaussian function is not the appropriate function. Second, noise has too much influence on covariance matrices of small objects (top right in the image). Changing the area fraction c to 0.1 seems

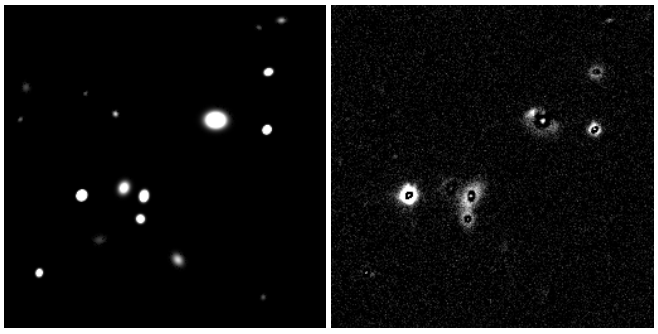


Fig. 12: Real image. Mixture model (left) and residuals (right). Area fraction $c = 0.1$. File `fpC-001231-r1-0109.fit` from [1].

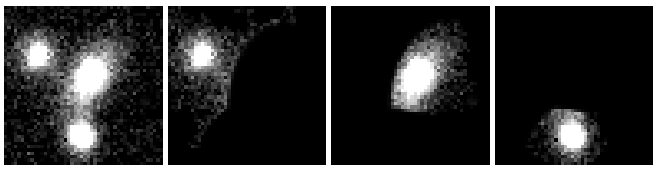


Fig. 13: Real input image (left) and extracted individual objects without modeling (right). File `fpC-001231-r1-0109.fit` from [1].

to reduce the problem in Fig. 11 (left). The residual image (Fig. 11, right) confirms that many objects are not correctly modeled.

Alternatively, individual objects can be extracted without modeling by copying the final *proto* values in Alg. 3 to a new image. Let p be the estimated probability that data points in a 1×1 window (determined by the pixel) are generated by that object. The *proto* values are image weights multiplied by p. Fig. 13 shows an example.

5 Discussion and future work

The result that 2D Gaussian functions do not model many objects in real images was expected. In future work, other models can be implemented by replacing the mvnpdf function in Alg. 3. Another possibility is to fit 2D functions to extracted individual objects in the algorithm.

The primary motivation of using the EM-algorithm for Gaussian mixtures was to try something new, instead of using Galfit. An advantage is that the implementation is simple and only pixels close to objects are used in computations. The benefit of using Galfit is that many models are available. In future work the methods can be compared.

Since there are some problems with the estimate of the covariance matrix, an estimate based on the pixels in the component is likely better than the area. This requires slightly more bookkeeping while computing attributes.

When MTObjects does not detect every object, variants of the EM-algorithm can still split a model of a single object into models of two objects. One variant is given in [19].

Including more color bands could improve results. One implementation is given in [10].

6 Conclusion

The combination of a max-tree and the EM-algorithm for Gaussian mixtures can be used to deblend 2D Gaussian functions in noisy images, although the number of plausibly modeled objects is limited by the number of objects detected by MTObjects. In the tested sample, object models are plausible even when the error is in the 90% quantile. Results showed that using the Mahalanobis distance, to discard pixels that are relatively far from objects, is essential to decrease the error. A Mahalanobis distance of 1.6 seems to be close to optimal.

When the number of iterations are fixed, the deblending method runs in $O(nm + n\log n)$ time, where n is the number of pixels, and m is maximum number of objects in a compound object.

In real images of the sky, a 2D Gaussian function does not model most objects, and other functions are required. More work is needed before the method can be applied to real images in sky surveys. Alternatively, it is possible to extract individual objects in the image without modeling, when using the mixture model as probability distribution.

7 Acknowledgements

We want to thank reviewers M. H. F. Wilkinson, J. Kwant, and M. al Shakoush.

References

[1] K. N. Abazajian et al. The seventh data release of the sloan digital sky survey. *The Astrophysical Journal Supplement Series*, 182(2):543, 2009.

[2] C. Berger, T. Géraud, R. Levillain, N. Widynski, A. Baillard, and E. Bertin. Effective component tree computation with application to pattern recognition in astronomical imaging. In *2007 IEEE international conference on image processing*, volume 4, pages IV–41. IEEE, 2007.

[3] E. Bertin and S. Arnouts. Sextractor: Software for source extraction. *Astronomy and astrophysics supplement series*, 117(2):393–404, 1996.

[4] H. Flewelling. Pan-starrs data release 2. In *American Astronomical Society Meeting Abstracts# 231*, volume 231, pages 436–01, 2018.

[5] S. Gazagnes and M. H. F. Wilkinson. Distributed connected component filtering and analysis in 2d and 3d tera-scale data sets. *IEEE Transactions on Image Processing*, 30:3664–3675, 2021.

[6] C. Haigh et al. Optimising and comparing source-extraction tools using objective segmentation quality criteria. *Astronomy & Astrophysics*, 645:A107, 2021.

[7] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2, page 236–243. Springer, 2009.

[8] B. Häussler et al. Gems: galaxy fitting catalogs and testing parametric galaxy fitting codes: Galfit and gim2d. *The Astrophysical Journal Supplement Series*, 172(2):615, 2007.

[9] C. E. Leiserson and B. M. Maggs. Communication-efficient parallel algorithms for distributed random-access machines. *Algorithmica*, 3(1):53–77, 1988.

[10] T. X. Nguyen et al. Object detection with component-graphs in multiband images: Application to source detection in astronomical images. *IEEE Access*, 9:156482–156491, 2021.

[11] G. K. Ouzounis and M. H. F. Wilkinson. Mask-based second-generation connectivity and attribute filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):990–1004, 2007.

[12] C. Y. Peng, L. C. Ho, C. D. Impey, and H. W. Rix. Galfit: Detailed structural decomposition of galaxy images. *Astrophysics Source Code Library*, pages ascl–1104, 2011.

[13] D. J. Prole, R. F. J. van der Burg, M. Hilker, and J. I. Davies. Observational properties of ultra-diffuse galaxies in low-density environments: field udgs are predominantly blue and star forming. *Monthly Notices of the Royal Astronomical Society*, 488(2):2143–2157, 2019.

[14] P. Salembier, A. Oliveras, and L. Garrido. Antiextensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7(4):555–570, 1998.

[15] P. Teeninga, U. Moschini, S. C. Trager, and M. H. F. Wilkinson. Statistical attribute filtering to detect faint extended astronomical sources. *Mathematical Morphology-Theory and Applications*, 1(1), 2016.

[16] P. D. Teeninga. Fast and memory efficient sequential max-tree construction using a trie priority queue. Technical report, University of Groningen, 2023.

[17] P. D. Teeninga. Parallel max-tree attributes in polylogarithmic expected time. Technical report, University of Groningen, 2023.

[18] P. D. Teeninga. Parallel max-tree construction in polylogarithmic expected time. Master's thesis, University of Groningen, 2023.

[19] D. Ververidis and C. Kotropoulos. Gaussian mixture modeling by exploiting the mahalanobis distance. *IEEE transactions on signal processing*, 56(7):2797–2811, 2008.

[20] M. H. F. Wilkinson. A fast component-tree algorithm for high dynamic-range images and second generation connectivity. In *2011 18th IEEE International Conference on Image Processing*, pages 1021–1024. IEEE, 2011.

university of groningen / faculty of science and engineering / computing science