FAIRNESS IN RECOMMENDER SYSTEMS: GRAPH-BASED APPROACH TO REDUCE THE POPULARITY BIAS

EVA ENGEL

Bachelor Thesis

Bachelor of Science (B.Sc.)
Department of Mathematics and Computer Science
Chair of Software and Systems Engineering
University of Cologne

August 2021

Eva Engel: *Fairness in Recommender Systems: Graph-Based Approach to Reduce the Popularity Bias,* © August 2021

MATRICULATION NUMBER:
7336646

SUPERVISORS:
Prof. Dr. Andreas Vogelsang
Patrick Ebel

EXTERNAL SUPERVISOR:
Marcel Kurovski (inovex GmbH)

LOCATION:
Cologne

DECLARATION

I hereby declare that the thesis submitted is my own, unaided work, completed without any unpermitted external help. Only the sources and resources listed were used.

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

Cologne, August 2021

Eva Engel

ABSTRACT

Recommender systems have become an indispensable part of web systems. They prevent their users from being overloaded with a flood of information by suggesting relevant items to them based on prior user interactions. However, recommender systems frequently face popularity bias issues: popular items are over-represented in the recommendations while less popular ones get less exposure. Consequently, the popularity bias not only threatens the fairness of recommendation results but also decreases the degree of personalisation of recommendations as popularity is not a direct indicator of personal relevance. Since the feedback loop further amplifies the popularity bias, debiasing strategies need to be developed to provide a fair and diverse recommender system.

The proposed approach identifies and reduces the popularity bias based on a predefined fairness policy. To do this, we modify PageRank, a graph-based algorithm developed by Google Search, which finds relevant users in a network. Fairness is integrated locally by adjusting the transition probabilities of a random walk on a graph according to the fairness policy. Depending on the desired fairness goal measured by the generalised Gini index, the results of this work present suitable fairness policies. In order to evaluate the method, experimental results are presented comparing the applied modifications to the unconstrained PageRank. The experiments are based on a public data set from the online social network Twitter.

Keywords – Popularity bias, recommender system, PageRank

ZUSAMMENFASSUNG

Empfehlungssysteme sind zu einem unverzichtbaren Bestandteil von Websystemen geworden. Sie sorgen dafür, dass die Nutzenden nicht mit Informationen überschüttet werden, indem sie ihnen auf Grundlage früherer Nutzendeninteraktionen relevante Artikel vorschlagen. Allerdings sind Empfehlungssysteme häufig von dem Problem der Popularitätsverzerrung betroffen: Beliebte Artikel sind in den Empfehlungen überrepräsentiert, während weniger beliebte Artikel weniger Aufmerksamkeit erhalten. Dies gefährdet nicht nur die Fairness der Empfehlungsergebnisse, sondern verringert auch die Personalisierung der Empfehlungen, da die Beliebtheit kein direkter Indikator für die persönliche Relevanz der Artikel ist. Da die Feedback-Schleife die Popularitätsverzerrung verstärkt, müssen Strategien entwickelt werden, die diese Effekte reduzieren, um ein faires und diverses Empfehlungssystem zu gewährleisten.

Wir haben einen Ansatz entwickelt, der die Popularitätsverzerrung identifiziert und anhand von vordefinierten Fairness-Richtlinien reduziert. Dieser Ansatz modifiziert PageRank, einen von Google Search entwickelten graphenbasierten Algorithmus, der relevante Nutzende in einem Netzwerk ermittelt. Fairness wird lokal integriert, indem die Übergangswahrscheinlichkeiten einer Irrfahrt auf einem Graphen entsprechend der Fairness-Richtlinien angepasst werden. Die Fairness von PageRank wird durch den verallgemeinerten Gini-Index gemessen. Je nach angestrebter Fairness stellen die Ergebnisse dieser Arbeit geeignete Fairness-Richtlinien vor. Um die Methode zu evaluieren, werden experimentelle Ergebnisse präsentiert, die die Modifikationen mit dem uneingeschränkten PageRank vergleichen. Die Experimente basieren auf einem öffentlichen Datensatz vom sozialen Online-Netzwerk Twitter.

ACKNOWLEDGMENTS

This bachelor thesis was written at the University of Cologne in cooperation with inovex GmbH, an IT project centre.

I want to express my gratitude to Prof. Dr. Vogelsang and Patrick Ebel from the Chair of Software and Systems Engineering at the University of Cologne for their kind supervision and collaboration.

A special thanks goes to the inovex GmbH for giving me the amazing opportunity to work on this topic. I want to especially thank my dedicated supervisor Marcel Kurovski. Thanks for all your guidance, your ideas, and for inspiring me with your extraordinary enthusiasm for Recommender Systems. Moreover, I want to thank my colleagues at inovex for their valuable feedback and especially Dr. Lea Petters for her multi-faceted mentoring.

Finally, I want to thank my family and my friends for their support during the whole time. I am extremely thankful for everything you have done for me.

CONTENTS

1 INTRODUCTION 1
- 1.1 Motivation 1
- 1.2 Research Question and Objective 3
- 1.3 Course of Investigation 4

2 BACKGROUND 5
- 2.1 Recommender Systems 5
  - 2.1.1 Definition 5
  - 2.1.2 Recommendation Techniques 6
- 2.2 Fairness and Biases 7
  - 2.2.1 Fairness Policy 7
  - 2.2.2 Biases 8
  - 2.2.3 Popularity Bias 8
  - 2.2.4 Generalised Gini Index 9
- 2.3 PageRank 10
  - 2.3.1 Fundamentals of Graph Theory 11
  - 2.3.2 Random Walks on Graphs 12
  - 2.3.3 PageRank for Web Search 14
  - 2.3.4 PageRank for Online Social Networks 15
  - 2.3.5 Challenges 16

3 STATE-OF-THE-ART AND RELATED WORK 19
- 3.1 Popularity Bias in Recommender Systems 19
- 3.2 Modifications of PageRank 20
- 3.3 Engagement Prediction 21
- 3.4 Fairness-Aware Link Analysis 22

4 PROPOSED METHOD AND IMPLEMENTATION 25
- 4.1 Data Set 25
  - 4.1.1 Online Social Network Twitter 26
  - 4.1.2 RecSys Challenge 2021 26
- 4.2 Data Preprocessing 28
  - 4.2.1 Graph-Based Representation of Twitter's Network 28
  - 4.2.2 Sampling 30
- 4.3 Fairness-Aware PageRank 32
  - 4.3.1 NetworkX - PageRank 32
  - 4.3.2 Fairness-Aware Transition Matrix 33

5 RESULTS AND DISCUSSION 37
- 5.1 Group Classification 37
- 5.2 Fairness 38
- 5.3 Comparison Fairness and Utility 40
- 5.4 Discussion 43

6 CONCLUSION AND OUTLOOK 45
A APPENDIX 47
A.1 Features of the Data Set provided by the RecSys Challenge 47
BIBLIOGRAPHY 49

LIST OF FIGURES

Figure 1.1	Twitter's long-tail	3
Figure 2.1	Feedback loop	6
Figure 2.2	Biases in the feedback loop	8
Figure 2.3	Lorenz curve	10
Figure 2.4	Dangling node coloured in red	17
Figure 4.1	Engagement graph	28
Figure 4.2	Breadth-first search to construct the engagement graph	29
Figure 4.3	Sample: Popularity distribution	31
Figure 4.4	Outgoing edges of a node in the graph	33
Figure 4.5	Modifcation: Node is connected to nodes of all groups.	34
Figure 4.6	Modification via paths in the neighbourhood	35
Figure 5.1	Lorenz curve for 5 bins	38
Figure 5.2	Lorenz curve for 100 bins	39
Figure 5.3	Parameters generated by linear functions	41
Figure 5.4	Colour gradient visualises fairness	41
Figure 5.5	Parameters generated by exponential functions	42
Figure 5.6	Utility loss compared to fairness	42

LIST OF TABLES

Table 4.1	Description of the training data set	27
Table 4.2	Possible seeds for personalised PageRank	31
Table 4.3	NetworkX Implementation	32
Table 5.1	Quantile-Based Discretization	37
Table 5.2	Largest impacts after applying the fairness-aware PageRank	39
Table A.1	Tweet Features	47
Table A.2	User features	48
Table A.3	Engagement features	48

ACRONYMS

AP	Average Precision
BFS	Breadth-First Search
CF	Collaborative Filtering
FAPR	Fairness-Aware PageRank
FSPR	Fairness-Sensitive PageRank
GGI	Generalised Gini-Index
LFPR	Locally Fair PageRank
OSN	Online Social Network
PR	PageRank
PR-AUC	Precision-Recall Area Under Curve
RS	Recommender System
RCE	Relative Cross Entropy
RMSE	Relative Mean Squared Error

1 INTRODUCTION

1.1 MOTIVATION

The invention of the world wide web has changed the nature of social interactions tremendously. Digitisation transcends geographical boundaries, turning the highly interconnected globe into a global village. People connect and share information all over the world, driven in part by online social networks. Our customary interactions, such as expressing ourselves or sharing ideas, are shifted to the digital realm. While using online social networks to spread and retrieve information, we generate vast and diverse data. The resulting global source of information has grown in terms of variety, volume and velocity.

Online social networks offer their users a platform to express themselves or join communities to exchange ideas. The variety and diversity of topics seem unlimited taking into account the different ways of thinking among people, both culturally and individually. A well-known social platform is Twitter, which recorded 199 million monetisable daily active users[1] in the first quarter of 2021 [51]. Even more impressive are Facebook's statistics for the same period – 1.88 billion daily active users[2], which corresponds to an increase of 8% compared to the previous year [22]. These figures illustrate the vast volume and potential of information that is generated by users. In years to come, the projected audience will be even larger with regard to the increasing number of users worldwide.

Besides the variety and volume of interacting users, velocity also features in social data. The quick and easy accessibility of social networks amplifies rapid spread and thus trends, reflecting the vibrancy of social media. Pew Research Center analysed the usage of the hashtag *#BlackLivesMatter* in publicly available Twitter tweets [8]. The Black Lives Matter movement first emerged in 2013 to protest against racially motivated violence against black people [38]. When the black US citizen George Floyd was murdered under police custody in 2020, the movement sparked again and widely spread on the Internet under the hashtag *#BlackLivesMatter*. One day after his death there were 218, 000 tweets containing this hashtag – three days later, May 28th, it

1 "Twitter defines monetizable daily active usage or users (mDAU) as people, organizations, or other accounts who logged in or were otherwise authenticated and accessed Twitter on any given day through twitter.com or Twitter applications that are able to show ads." - Twitter [51]

2 "We define a daily active user as a registered Facebook user who logged in and visited Facebook through our website or a mobile device, or used our Messenger application (and is also a registered Facebook user), on a given day." - Facebook [22]

was used almost 8.8 million times. This important event demonstrates the power of online social networks as well as the velocity of social interactions and networking over the internet.

It can thus be concluded that social platforms generate data that is not only diverse and large because of the variety of users, but also emerging and changing. One might assume that users generally benefit from a wide range of content. However, the amount of available information can result in an information overload, which describes the state in which users are not able to effectively process and utilise it to make decisions. Consequently, online social networks and the world wide web, in general, pose new challenges to information retrieval. In order not to overload users, relevant content must be filtered. A *recommender system* deals precisely with this task: it recommends relevant content. Since relevance is construed on a subjective, personal level, recommender systems take user preferences into account to provide individualised information. The personalised practice of recommender systems is found in many everyday situations – when looking for products online or streaming multimedia, for instance. Companies like Amazon, Spotify or Netflix use them to personalise content with the aim of increasing personal relevance and thus retaining customers.

To generate recommendations, recommender systems mainly rely on both implicit and explicit feedback provided by users consuming the particular web system. In other words, recommender systems assess which content is relevant based on the users' behaviour, which can consist of a like of particular content or a purchase of an offered item, for instance. A collaborative filtering approach identifies the similarities between users based on their behaviour in order to make recommendations. This approach tends to suffer from a popularity bias [34]: popular content gets a lot of exposure, while less popular content is under-represented in the recommendations. Popular content thus gets the opportunity to receive more feedback and be deemed of higher relevance. Therefore, recommender systems tend to suggest popular content to the users more frequently if this is not taken care of. Due to the feedback loop, in which recommendations themselves, in turn, receive feedback, popularity poses a self-reinforcing mechanism. Less popular content remains less popular and the problem persists.

The effect of popularity bias is especially crucial for recommender systems in online social networks. Popular users have a wide audience, which allows them to receive more feedback. The recommender system rates the content as more relevant, which in turn means they get more exposure on the platform. However, social platforms are primarily represented by their low-reach users, who nurture the exchange. Figure 1.1 illustrates Twitter's popularity distribution in terms of the number of followers. The popular users with more than 1500 followers only account for a small percentage of all users. In contrast, the majority, who are in the long-tail of the distribution, can be considered as unpopular.

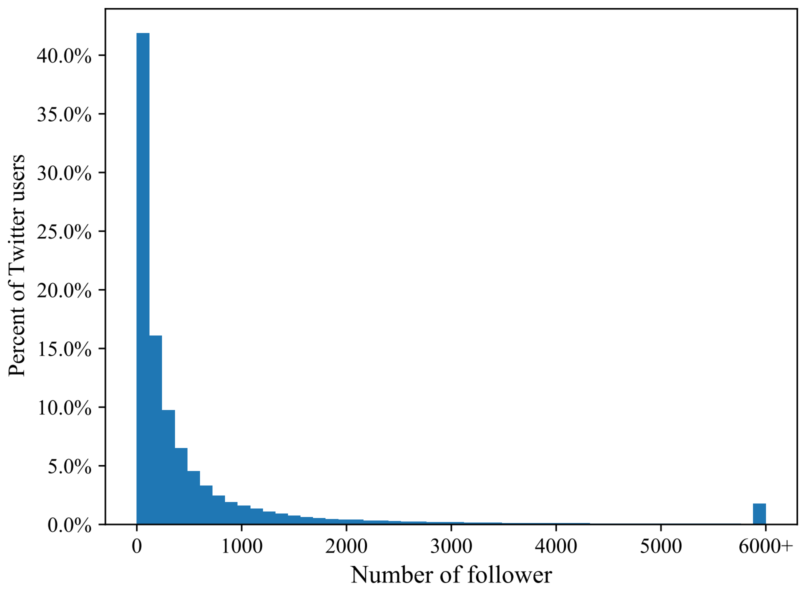

Figure 1.1: Twitter's long-tail

Although these users shape social platforms in the first place, they are deprived of the possibility to be heard. Furthermore, popularity is not a direct indicator of content relevance, especially when recommendations should be personalised. Consequently, social-based recommender systems should be aware of the popularity bias and incorporate measures to reduce its influence in order to develop a fairer recommender system.

1.2 RESEARCH QUESTION AND OBJECTIVE

The main objective of this thesis is to develop a debiasing strategy to reduce the popularity bias. Therefore, we investigate the graph-based algorithm PageRank – developed by Google Search – quantifying how important and influential a specific node is to its network.

As a step toward this goal, we will first investigate the concept of fairness in recommender systems. Second, we will develop an approach to modify the PageRank algorithm in a way that diminishes the popularity bias. Finally, we will evaluate and optimise the applied adjustments.

We turn the research objectives into three research questions, which will serve as a guideline for this work:

RQ1. What does fairness in a recommender system mean and how can we quantify it?

RQ2. How can we adjust the PageRank algorithm to reduce the adverse effect of user popularity?

RQ3. How do we optimise the trade-off between PageRank's utility and fairness?

1.3 COURSE OF INVESTIGATION

In order to answer the research questions of this thesis, we will pursue the following course of investigation. In Chapter 2, we start by giving an overview of the theoretical concept of a recommender system along with the fairness associated with its recommendations. Here, the first research question RQ1 will be addressed. Then, we will provide the mathematical fundamentals to delve deeper into the PageRank algorithm. Chapter 3 presents the state-of-the-art of fairness in recommender systems as well as modifications for the PageRank algorithm. In Chapter 4 we will describe our proposed method and its application to the data set. Thereby, we answer research question RQ2. Chapter 5 serves to evaluate and discuss our approach. We will focus on the optimisation problem regarding research question RQ3. Finally, Chapter 6 summarises the work and shows the impact of this thesis on the field of research. Furthermore, we will give a short outlook on further ideas in this area of research.

2 BACKGROUND

This chapter provides the background to understand the popularity bias in a Recommender System (RS) and the proposed debiasing method based on PageRank (PR). First, Section 2.1 gives an overview of the concept of RSs. In Section 2.2, we will address fairness and biases in the context of web-based applications including RSs and explain the Generalised Gini-Index (GGI). Finally, we will present the mathematical principles to explain and derive PR.

2.1 RECOMMENDER SYSTEMS

2.1.1 *Definition*

RSs are software tools and techniques suggesting items that are most likely of interest to a particular user [42]. *Item* is generally referred to as what the system recommends to a user. The recommendations relate to various decision-making processes, for example, which movie to stream, which products to buy or which content to watch on social media.

In order to provide recommendations, RSs require prior information about the users' preferences. This information builds on user *feedback*, which can be separated into two types: *implicit* and *explicit* feedback. Explicit feedback is obtained when the user actively expresses her or his preference for an item. There are several ways to collect explicit feedback from users. A web application may ask users to rate items according to a "like"/"dislike" function or according to a discrete numerical scale indicating how much the user liked or disliked the content [7]. In contrast, implicit feedback does not require user participation. It is based on user actions, such as which items they visited or how long they stayed on a web page. Implicit feedback is thus derived from the user's actions, which are not primarily intended to signal a preference [42].

When a RS provides recommendations to users, the interaction of the three components – user, data and model – can be abstracted in a *feedback loop* [35]. Figure 2.1 illustrates the feedback loop which consists of three resulting phases. When a user interacts with a system, it generates data, which is collected and supplemented with further side information (e.g. user profiles, item attributes). Depending on the filtering approach of the RS, the generated data is processed differently. The model learns based on this data. Subsequently, the RS provides recommendations to the users to satisfy the users' need for

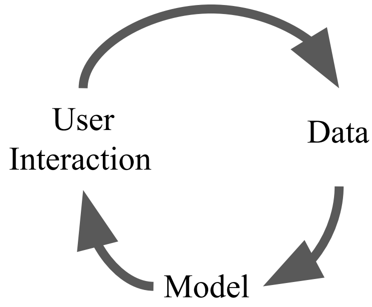

Figure 2.1: Feedback loop with the three key components model, data and user interaction

information. The recommendation affects the behaviours and decisions of the users, which in turn affects the data and thus creates a loop.

2.1.2 *Recommendation Techniques*

There are various approaches to develop a RS [42]. The most common techniques are *Content-Based Filtering* and *Collaborative Filtering (CF)*. It is possible to combine different approaches which results in a *hybrid RS*.

A content-based RS learns to recommend items based on the similarity of item attributes. For example, if a user has positively rated a horror movie, the system can learn to recommend other movies from this genre. Content-based techniques aim to match the attributes of the generated user profile with the item attributes.

CF provides recommendations by analysing relationships between users. This can be done by comparing the items they have explicitly rated in the past. The core of this, so called neighbourhood-based approach is to find like-minded users [20]. There are two general classes of CF: memory-based and model-based [48]. Memory-based algorithms make predictions by loading the entire or a sample of the user-item database. The data is stored in the form of a rating matrix that contains all item ratings provided by users. Based on this rating matrix, the algorithm determines the most similar user vector to find the nearest neighbours and like-minded users. This procedure is simple but encounters the problem of data sparsity. Users usually interact with only a fraction of the items of a web system [48]. In contrast, model-based approaches utilise the user-item database to learn a model, which provides rating predictions. Common approaches, which are further described in [20], are clustering, classification, latent model, Markov decision process, and matrix factorisation.

2.2 FAIRNESS AND BIASES

In this section, we will outline fairness and biases in the context of web-based applications, such as RSs. We will describe the different sources of bias, with a special focus on the popularity bias. Finally, we will introduce the GGI, a metric that measures the inequality of a distribution, allowing us to quantify the popularity bias.

2.2.1 *Fairness Policy*

Whenever a system needs to allocate a limited amount of resources (e.g. a ranking) among several individuals or groups, the concept of *fairness* arises [19]. A common definition of fairness is "absence of any prejudice or favoritism towards an individual or a group based on their intrinsic or acquired traits" [35]. Thus, fairness is a multi-faceted concept that needs to be assessed from different perspectives. A distribution policy that seems fair to one party may not be similarly reasonable for the other parties.

A *fairness policy* describes the distribution of a limited amount of resources R to individuals $v \in V$ [50]. We define R as a normalised ranking function that assigns a value between 0 and 1 to individuals. Furthermore, the set of individuals is divided into k disjoint groups:

$$\dot{\bigcup}_{i \in (1,\dots,k)} V_i = V \tag{2.1}$$

An individual $v \in V$ has a node attribute $i \in (1, ..., k)$ that describes the group V_i to which they belong. A fairness policy for R is defined as:

$$R[V_i] = \phi_i, \quad \text{for} \quad i \in (1, ..., k) \quad \text{with} \quad \sum_{i=1}^{k} \phi_i = 1 \tag{2.2}$$

Fairness policies can be included in an algorithm *globally* as well as *locally* [50]. Global fairness means that the defined fairness policy has to be guaranteed on average across a group and not on each node. Thus, the computational complexity is lower and one could theoretically guarantee the fairness conditions with k steps: One could select one node from each group V_i and adjust the k weights to meet the given policy. It is also easier and more precise to achieve arbitrary fairness policies, as we do not have to fulfil them at every node. In contrast, local fairness requires a fair allocation for each node in the graph. Furthermore, fairness-aware approaches can be divided into *pre-processing*, where the input data is changed, *in-processing*, where the algorithm is changed, and *post-processing*, where the output is changed [50].

2.2.2 *Biases*

When designing and engineering web systems, it is important to take fairness issues into consideration. As web systems are used in many sensitive environments to support users and assist in decision-making, they should address users' needs without reflecting *biases* and discriminatory behaviour.

RS or web systems, in general, are vulnerable to biases that render their outputs unfair [9]. CF-generated recommendations generally suffer from bias towards certain groups of users or items [34]. These biases can be categorised and defined based on the phases of the above-defined feedback loop. Figure 2.2 illustrates several biases which refer to the arrows and thus to the phases of the feedback loop. Since not every bias can be clearly assigned to one of the phases, various classifications can be found in the literature.

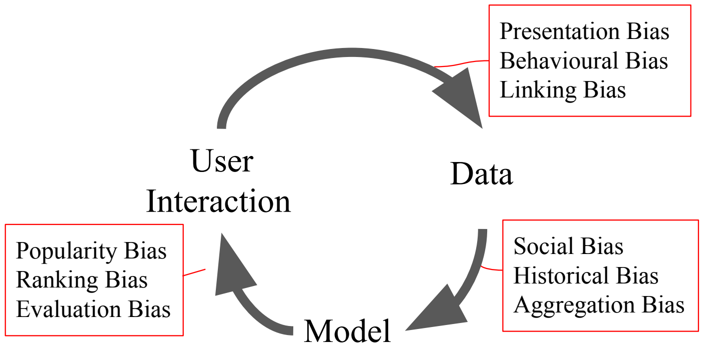

Figure 2.2: Biases in the feedback loop based on the components model, data and user interaction

2.2.3 *Popularity Bias*

Popularity bias is a problem that frequently occurs in RSs [16, 34]. Thereby, the so called long-tail phenomenon plays an important role in the data generated by user interactions. A small portion of popular items accounts for the majority of user interactions. When the RS is trained with such long-tailed data, it usually distributes higher scores for popular items than for unpopular items. As a result, popular items are recommended more frequently [16]. Thus, the imbalance in usage data is propagated and the popularity bias is amplified over time due to the feedback loop [34].

Popularity is an easily accessible measure of success that can be based on the number of streams of a particular movie or the number of followers of a social media user, for example, [17]. The effectiveness of such metrics relies on *wisdom of the crowd*: "large groups of people are smarter than an elite few" [15]. A individual's behaviour may be influenced and guided by the decisions of peers or neighbours, as popularity-based data is often taken as input for future recommendations. However, these metrics are subject to manipulations, such as fake reviews or fake accounts, known as social bots. Social bots can post content and interact on social media, which is a low-cost way to influence public opinion [45].

If popularity is not taken into account, this leads to problems [16]. To begin with, the level of personalisation decreases as popular items would be recommended more intensively. Although popular content is popular for legitimate reasons, it is not necessarily relevant to all users, especially as user preferences vary. Another problem is that the popularity bias threatens the fairness of recommendation results. Over-recommending popular items reduce the exposure of other items, even if they would be a good match.

As a result, RS should be aware of the popularity bias. quality is not necessarily correlated with popularity [17]. First, it is possible to maintain a good correspondence between popularity and quality rankings of consumed items even when our reliance on popularity for our choices is relatively high. Second, one can leverage the wisdom of the crowd in the presence of limited attention, or let users make their own decisions when they are able to explore many items

In addition to the popularity bias, there are various biases that can occur. Since this work focuses on the popularity bias, we will outline two further biases that occur more frequently in Online Social Network (OSN)s (the other biases are detailed in [9]). The presentation bias results from the way information is presented. Users can only interact with content that they see. This bias has a direct effect on new content or content that have never been seen before, as there is no usage data yet. A common solution is the explore/exploit scheme by Agarwal, Chen, and Elango. A portion of users is shown new items that are randomly assigned with top recommendations. Thus, these items are explored and, if interacted with, the usage data is exploited to determine their true relative value.

Social bias defines how feedback from other people influences the user's judgement. Considering collaborative ratings, a user wants to rate an item with a low score but sees that most users have already given it a high score. The user might get influenced by the other users and increase the rating thinking that she/he might be too harsh. This phenomenon is often referred to as "social conformity," or "the herding effect".

2.2.4 *Generalised Gini Index*

The process of generating recommendations obtains many sources for the occurrence of biases that jeopardise fairness and influence the relevance of recommendations. It is crucial to identify these biases and preferably measure the degree of fairness in a RS. Therefore, we will present the GGI in order to investigate the popularity bias in the data set.

The GGI is a measure of distributional inequality, which describes the degree of fairness of an allocation policy [36]. It is defined in terms of the Lorenz curve $L(\rho)$, which is a probability plot comparing the distribution of a variable on the y-axis against its uniform distribution ranked for a specific attribute on the x-axis. The Lorenz curve has a positive slope and is piecewise (weakly) convex. Initially, Lorenz developed the curve to present the relationship between the cumulative distribution of income units ordered by their income [11]. Figure 2.3 illustrates the Lorenz curve for income distribution. The value $L(\rho)$ describes the total income earned cumulatively by the ρ% of the lowest income units.

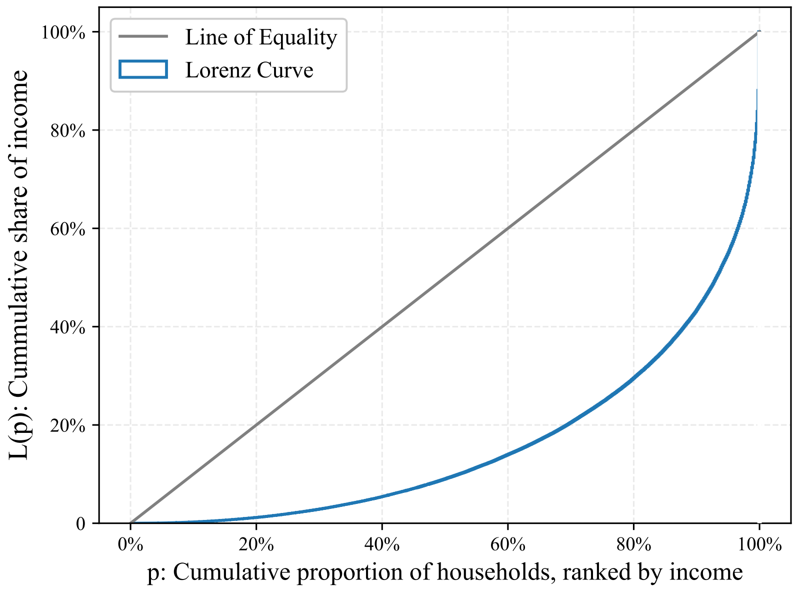

Figure 2.3: Lorenz curve for an income distribution

The GGI is calculated as twice the area between the Lorenz curve $L(\rho)$ and the line of perfect equality $f(x) = x$, the most equal distribution. Its values vary from 0 to 1, the higher the value and the less equal the allocation. Since in most cases the Lorenz curve is discrete, the area of the Lorenz curve is calculated approximately using numerical integration techniques (see [11] for further investigations).

2.3 PAGERANK

The following section introduces PR, the best-known algorithm for link analysis quantifying how important and influential a specific node is to its network [31]. In the first step, we will provide the required fundamentals of graph theory to clarify random walks on graphs which is the underlying principle and the centrepiece of PR. Next, we will outline both the original use of PR in web search as well as for online social networks. We will finish this section by introducing the challenges associated with the use of the PR algorithm.

2.3.1 *Fundamentals of Graph Theory*

A graph $G = (V, E)$ consists of two sets V and E. V is a finite, nonempty set of elements which are called *nodes* (or *vertices*). The possibly empty set E consists of *edges* which connect a pair of nodes $\{i, j\}$ with $i, j \in V$ or a node to itself. Let $N = |V|$ be the order of G, i.e. number of nodes, and $M = |E|$ the number of edges. A node i is *adjacent* to node j if they are connected by an edge. In this case, the two nodes are called *neighbours*. The set of neighbours of a node i is denoted by the *neighbourhood* $K(i)$.

The *adjacency matrix* A_{NxN} is representative for the graph and indicates whether pairs of nodes are adjacent or not. Furthermore, edges can be formulated in an *adjacency list* where the keys are the nodes and the values are the incident edges. Moreover, the degree $d(i)$ of i is the number of edges which contain i, i.e. $|K(i)|$.

To utilise graphs for mathematical modelling, nodes can have additional attributes beyond nodes and edges: A node attribute is defined as a function from the node-set to some set of possible attribute values [25]. Analogously, edge attributes are defined. An attribute often used is the weight $w_{ij} \in \mathbb{Z}$ of an edge $\{i, j\}$ [54].

A *digraph* (or directed graph) is a graph in which the edges are directed and assigned a pair of vertices (u, v). The node u is designated as the tail and v as the head. For a weighted digraph, the adjacency matrix is defined as follows:
$A = (a_{ij})_{NxN}$

$$a_{ij} = \begin{cases} w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

In a digraph, we distinguish between the *in-degree* and *out-degree* of a node [41]. The number of edges linking to a node v is called the in-degree and denoted as $d^{+}(v)$. Hence, $d^{-}(v)$ describes the out-degree of v, i.e. the number of edges leaving v.

A *path* of length $k \in \mathbb{N}_0$ is a non-empty graph $P = (V', E')$ of the form:

$$V' = \{x_0, x_1, ..., x_k\} \quad \text{and} \quad E' = \{\{x_0, x_1\}, ..., \{x_{k-1}, x_k\}\}$$

For simplicity, we refer to the path as the sequence of its nodes $P = (x_0, x_1, ..., x_k)$. The distance $dist(u, v)$ of two nodes u, v is the length of the shortest u-v path in G; we set $dist(u, v) := \infty$ if no such path exists.

H is a *subgraph* of the *supergraph* G if $V_H \subseteq V_G$ and $E_H \subseteq E_G$. A special case is the *induced subgraph*, which contains a subset of nodes $W = \{v_1, ..., v_k\} \subseteq V_G$ and necessarily all edges of G whose endpoints are in W.

Having stated some definitions, we will outline *Breadth-First Search (BFS)* which we need for generating the sample graph in Section 4.2.1. BFS is a classical graph traversal algorithm, which is characterised by visiting each node of the graph while avoiding visiting a node twice. Essentially, BFS starts from a seed node and progressively explores all neighbours. In the first iteration, the neighbours of the seed node are added to the end of the queue and marked as visited. At each new iteration, the node that was explored earliest but has not yet been marked is selected as the next node. In this way, BFS discovers all nodes on a level within a certain distance from the seed and then moves to the next level.

When the graph structure is saved in an adjacency list, i.e. in a dictionary format, Listing 2.1 shows an implementation of BFS.

Listing 2.1: Breadth-first search in Python, starting at node S

```python
def bfs(G, S): #function for BFS
  visited = [S]
  queue = [S]
  while queue:          # Creating loop to visit each node
    node = queue.pop(0)
    for neighbour in G[node]:
      if neighbour not in visited:
        visited.append(neighbour)
        queue.append(neighbour)
```

The time complexity of BFS is $O(N + M)$ when utilising an adjacency list and $O(N^2)$ for an adjacency matrix.

In comparison, *depth-first search* is based on the same principle as BFS but instead of using a queue, the nodes are saved in a stack. Thus, depth-first search follows paths and explores the network in depth.

2.3.2 *Random Walks on Graphs*

Random walks on complex networks have wide applications in a variety of scientific fields, leading to an important understanding of dynamic processes and network structures [54].

For a graph $G = (V, E)$ with $|V| = N$ nodes and $|E| = M$ edges, the adjacency matrix $A \in \mathbb{N}^{NxN}$ represents its connectivity. In this definition, we will restrict ourselves to weighted digraphs with non-negative weights $a_{ij} = w_{ij}$. However, random walks can be formulated on various graphs, such as unweighted graphs with $a_{ij} = 1$ if nodes i and j are adjacent or 0 if not. Moreover, the out-degree of node i is defined as $d_i = \sum_{j=1}^{N} a_{ij}$.

First, we derive the *transition matrix* $T \in \mathbb{R}^{NxN}$ from the weighted adjacency matrix A, which describes the transition probability of reaching the neighbouring nodes within one step:
$T = (t_{ij})_{NxN}$

$$t_{ij} = \begin{cases} \frac{w_{ij}}{d_i} & \text{if}(i,j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

Notice the bias of the transition probability occurring on a weighted network is proportional to the weight w_{ij} of the corresponding edge. By our definition, the transition matrix $T \in \mathbb{R}^{NxN}$ is square, non-negative and the entries t_{ij} are normalised such that the row sums equal 1.

A random walk on G is now defined as follows. Starting at an arbitrary node i, the walker follows an outgoing edge with the transition probability t_{ij} and reaches a node j in the neighbourhood $K(i)$. This statistical process can be described by a *Markov chain* [40], which is a random process where at any given time $t = 1, 2, 3 \ldots n$ one of the finitely many states in a system occurs. At each time t the system changes from state i, in this case node i, to node j with probability t_{ij}. Note that the probabilities are independent and that the Markov chain decides on the next state by considering only the current state.

With the mathematical basics covered, we now explain how random walks can generate information about the network structure [54]. Therefore, we want to determine $p_t \in \mathbb{R}^N$, the probability distribution of the Markov chain at time t. In other words, p_t describes the probability of the random walk being at a node after t steps for all nodes. To do this, we compute the transition matrix which we use to initiate random walks on the graph. Assuming we start equally distributed on all nodes with p_0, we get the following result for our first iteration $t = 1$:

$$p_0^\top = \left(\frac{1}{N}, \cdots, \frac{1}{N}\right) \tag{2.5}$$

$$p_1^\top = p_0^\top \cdot T = \left(\sum_{i=1}^{n} \frac{t_{i1}}{N}, \cdots, \sum_{i=1}^{n} \frac{t_{in}}{N}\right) \tag{2.6}$$

By applying the transition matrix T iteratively, the probability converges to p, a stationary distribution of the Markov chain. This means that p does not change after an additional iteration. p is independent of the starting distribution p_0 if the graph is connected in such a way that there is a path between all nodes (a more detailed investigation on the required conditions for a stationary distribution can be found in [37]). The resulting vector p provides information about the characteristics of the weighted graph and describes which nodes are more likely to be visited.

2.3.3 *PageRank for Web Search*

The heart of our software is PageRank.

Google, 2002

PR was first presented in 1998 with the release of the initial prototype of the *Google* search engine. In the original context of searching a web page, PR exploits the link structure of the *World Wide Web* to assign a numerical weight to each web page measuring their relative importance [37]. In other words, PR models the behaviour of a random surfer who browses the web and randomly clicks on an outgoing link from one page to go to the next one. As a result, PR's output is the probability of the surfer arriving at a particular page [23]. The more incoming links a page has, the more important the page is. Consequently, much-cited pages have a greater influence and their outgoing edges have more weight.

The link structure of the web, in which pages refer to each other via hyperlinks, can mathematically be abstracted as a graph. Nodes represent the web pages and the directed edges describe if one page links to others. Given a directed Graph $G = (V, E)$ with transition matrix T, PR returns a vector p which contains the score $\mathrm{PR}(u)$ for each node $u \in V$ in the graph. The score $\mathrm{PR}(u)$ for each node is between 0 and 1 stating the probability of the random walk being at the particular node. Thus, $\sum_{u \in V} PR(u) = 1$ and p has the natural normalisation:

$$p^T e = 1, \quad \text{where } e \text{ is a vector of all ones} \tag{2.7}$$

PR is based on a random walk (see Section 2.3.2), but with the additional possibility of restarting at each step. More specifically, the random walk follows either the transition matrix T with probability α (damping factor) or jumps to another node which is selected according to a so called jump vector v. Note that the random walk can jump to a node that is not connected via edges. The jump vector v defines a distribution over all nodes and is typically set to the uniform vector. However, a special case of PR is when the restart vector is defined as a unit vector $v = e_i$ that places the entire mass on a single node i. Then, we speak of a personalised PR that restarts at the fixed node i. In this case, the probability $\mathrm{PR}_i(u)$ approximates the proximity between the nodes u and i, which can be used for a RS.

Although Page and Brin do not specifically refer to the Markov chain

in their original PR algorithm, Langville and Meyer and Ravi Kumar, Alex Goh, and Ashutosh explored the connection between PR and the Markov chain. PR computes the unique stationary distribution for the modified Markov chain (more precisely irreducible, aperiodic chain [40]).
The PR scoring vector p satisfies the following equation:

$$p^T = \alpha \underbrace{p^T T}_{transition} + (1-\alpha) \underbrace{v^T}_{jump} \tag{2.8}$$

One way is to solve the above equation algebraically. The solution is given by:

$$p^T = (1-\alpha) v^T \, (\mathbb{1} - \alpha T)^{-1} \tag{2.9}$$

However, this approach is not efficient for a large sparse matrix T. Thus, we will introduce *power iteration* instead. Power iteration is a numerical algorithm for computing the dominant eigenvalue and its eigenvector [32]. As p is normalised (see Equation 2.7), we can transform the above Equation 2.8 to a eigensystem for the modified *Google* matrix M:

$$p^T = \alpha p^T T + (1-\alpha) p^T e v^T \tag{2.10}$$

$$\Leftrightarrow p^T = p^T \underbrace{(\alpha T + (1-\alpha) e v^T)}_{=:M} \tag{2.11}$$

We obtain the eigenvector problem $p^T = p^T M$. Since the largest eigenvalue of M is 1 [21], the power iteration is defined as follows:

$$p_{k+1}^T = \frac{p_k^T M}{||p_k^T M||} = p_k^T M \tag{2.12}$$

The method iterates until $|p_{k+1} - p_k| < \epsilon$ for a small tolerance ϵ. Then, $p = (PR(u_1) \cdots PR(u_N))^T$ is the solution for PR on graph G, characterised by the damping factor α, the error tolerance ϵ and the jumping vector v.

Recall that the damping factor $\alpha \in [0, 1]$ decides whether the random walk follows the transition matrix with probability α or jumps to another node. If α is chosen small, PR would converge faster, but the smaller α the less the actual hyperlink structure of the web is incorporated in determining the importance of the web page. Thus, the choice of the damping factor needs to be considered carefully regarding convergence and the degree of involving the hyperlink structure. The founders of Google, Brin and Page, propose $\alpha = 0.85$ [23].

2.3.4 *PageRank for Online Social Networks*

The concept of PR — determining influential web pages via the hyperlink structure — can be generalised to further graph-based information networks, such as social networks [31]. A social network can be defined as a graph $G = (V, E)$ where nodes represent individuals (or organisations) and the edges represent their social relations. Specifically for an OSN, users are connected through bilateral or unilateral relations. Besides conventional mutual friendships, users can follow other users to be informed of their content. For simplification, we speak of users, even though an account on a social platform could represent an organisation or company, for instance.

Another possibility to create a graph is based on their interactions and their number of interactions. Thus, the digraph can consist of weighted edges, that influence the transition probabilities of the random walk. This modification can be achieved by adjusting the transition matrix T proportionally to the weight. Apart from that, the PR algorithm proceeds in the usual way and computes a scoring vector that describes the importance and influence of a user in the OSN.

PR and especially personalised PR can be employed for various user-centred applications. One field deals with link prediction and RS design. The social structures of the network are used to determine possibly relevant users and indirect their content [33].

2.3.5 *Challenges*

In real-world applications, we find graphs that would not converge or give a reasonable result when calculating PR. Therefore, we need to identify these potential cases and then mostly modify the equation using the transition probabilities.

A common problem is caused by *dangling nodes* [40]. A dangling node does not have any outgoing edges which is equivalent to $d^{-}(v) = 0$. Figure 2.4 illustrates a dangling node, coloured in red. When a random walk encounters a dangling node, it is not clear how the weights should be distributed for a transition to a neighbouring node. Thus, a dangling node keeps accumulating more and more PR weight whenever it is visited. Furthermore, the transition matrix T has a row containing all zeroes which leads to T not being stochastic and the random walk not being presentable in a Markov chain model [32].

One way to handle the dangling node problem is to add transition probabilities and thus temporally create edges outgoing from the dangling node. Langville and Meyer proposed a method to allocate the weights from a dangling node according to a fixed distribution d. The distribution d is defined by a personalised vector which is, for instance, uniformly distributed over all nodes [32]. For the Equation 2.8, we set w_{Nx1} with $w_i \in \{0, 1\}$, which indicates whether node i is dangling or not, and get the modification:

$$p^T = \alpha p^T (T + wd^T) + (1 - \alpha) v^T \tag{2.13}$$

Another challenge is the sparsity of the transition matrix [32]. Although not all nodes are connected, the transition matrix T_{NxN} is a

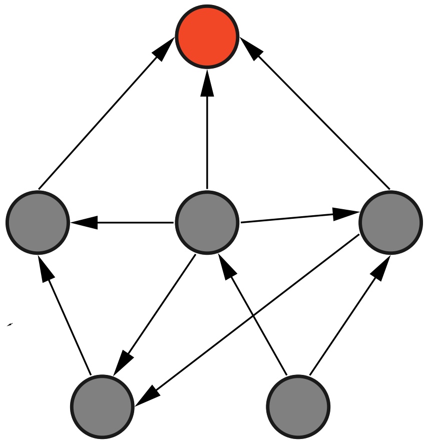

Figure 2.4: Dangling node coloured in red

square matrix defined by the number of nodes. This can lead to memory and computational issues, especially when decomposing a matrix. Regarding the power iteration, it can exploit the sparsity of the matrix for the matrix and vector multiplications. However, if transition probabilities are now artificially added to solve the dangling node problem, for instance, the sparse property decreases.

3 STATE-OF-THE-ART AND RELATED WORK

The research questions of this thesis specifically address the two research areas, fairness in RS and modifications of PR. These two research fields differ not only in their purposes but also in their history and development. The discussion of fairness in RS sparked in the 2010s as the use of RS increased. In 2013, Bozdag addressed biases in algorithmic filtering and expanded the idea of technical biases to human and social biases [15]. The interest in algorithmic fairness has increased significantly in recent years - especially in the context of machine learning. On the other hand, research of PR has gained importance in the 2000s due to the success of Google. There have been many approaches to improve the efficiency of the algorithm or to apply the idea to other concepts and applications.

First, we will outline related work on fairness in RS and modifications of PR. Subsequently, we will highlight two research papers that sparked our research. In Section 3.3, we will present the research of Goda, Agata, and Matsumura that propose an implementation of an RS to predict engagements for the social network Twitter. They use PR to generate further features for their model to represent users' social influence and extract user similarities. Section 3.4 will highlight the work of Tsioutsiouliklis et al. who devise approaches for a Fairness-Aware PageRank (FAPR).

3.1 POPULARITY BIAS IN RECOMMENDER SYSTEMS

The concept of fairness in RSs has been gaining a lot of attention in the past years [1]. Among other issues, popularity bias has been the subject of many research studies [5, 16, 47].

Mansoury et al. studied the impact of the feedback loop on the popularity bias amplification of several RSs [34]. To do this, they simulated the users' interaction with a RS in an offline setting. The results additionally revealed two potential problems which arise due to bias amplification. It shifts the representation of the user's preferences over time and creates a homogenisation of user groups. In the same context, Abdollahpouri introduced the metric Algorithmic Popularity Lift measuring the amplification in the generated recommendations of a RS for different user groups [1]. In order to deconvolve feedback loops, Sinha, Gleich, and Ramani proposed a method based on the assumption that feedback has a certain mathematical relationship with previous recommendations [46]. Research on this topic emphasises the importance of algorithmic solutions to address the popularity bias. Even a minor bias in the current state of a RS can be significantly amplified within iterations if not handled appropriately [34].

Several debiasing strategies have been developed that aim to recommend items from the long-tail [3, 29]. The regularisation-based approach in [2] provides a flexible framework for fairness-aware optimisation built on a learning-to-rank algorithm. However, their approach is in-processing and thus restricted to factorisation models that include long-tail preferences as a latent factor. The work of Abdollahpouri, Burke, and Mobasher, instead, introduced a personalised reranking approach for long-tail promotion, which can be deployed on the output of any RS [4]. Among other metrics, they used Average Recommendation Popularity and Average Percentage of Long Tail Items to evaluate the effectiveness of algorithms in reducing the popularity bias [4]. And finally, Sun, Shafto, and Sun proposed a matrix blind-spot-aware matrix factorization algorithm to debias the interaction between users and RSs [47]. To do this, they calculated both the Relative Mean Squared Error (RMSE) to evaluate the accuracy of the prediction and the GGI to test the debiasing mechanism.

3.2 MODIFICATIONS OF PAGERANK

A lot of research has been done on the efficiency and performance of PR in the regular context of web pages [28, 32]. Recent work in this field has shifted to the study of various generalisations of PR, such as for OSNs [10, 44].

To begin with, Xing and Ghorbani extended the PR method to weighted graphs [52]. The rank of a web page is not evenly distributed among the outgoing links but instead, it is distributed proportionally to the outgoing links' popularity. Web page popularity is based on the number of in- and outgoing links and thus included of the network structure and no additional information. The work presented in [43] deals with debiasing the link farm effect in PR. Since websites are strongly interested in a high ranking in the search engine results, attempts are made to artificially boost the ranks by spamming the link structure of the web. Therefore, they present a method to distribute the weight accumulated by link farms to other web pages by modifying the transition probabilities. The weights from the nodes that belong to a link farm are uniformly distributed to every other node in the web graph. The graph is constructed based on the Yahoo-API and consists of $250,000$ nodes [43].

Recent research on PR also includes machine learning approaches. Bogolubsky et al. proposed a gradient-free optimization method to learn supervised PR, which accounts for nodes and edges attributes [12]. The work of Bojchevski et al. utilises a graph neural network to classify nodes. The model scales to large graphs by using an adjusted propagation approach based on the approximate personalised PR. A personalised PR indicates the information of a neighbourhood of a given node, but cannot be easily applied to large graphs due to the expensive computation of power iteration and must therefore be approximated. They used two commonly used data sets as a benchmark: Cora-Full (18.7K nodes, 62.4K edges, 8.7K node features) and PubMed (19.7K nodes, 44.3K edges, 0.5K node features). In addition, they introduced the MAG-Scholar-C data set (10.5M nodes, 133M edges, 2.8M node features) [13].

In order to identify the issuer of a topic in a social network, Priyanta and Nyoman Prayana Trisna used the ranking result of PR and compared it to other approaches based on closeness centrality and the in-degree of a node. They used NetworkX to generate a network that illustrates $18,000$ tweets obtained with Twitter API [39].

Heidemann, Klier, and Probst utilised PR to identify key users based on their communication activity and connectivity in a OSN [26]. A key user can affect a large number of friends or other users in an OSN. Determining these users enables marketing campaigns to develop more effective advertising strategies and reach a wider audience. Therefore, they evaluated their approach on a social graph of Facebook containing $63,731$ users and $817,090$ undirected social links.

3.3 ENGAGEMENT PREDICTION

Goda, Agata, and Matsumura propose a stacking ensemble model to predict user engagements on Twitter. They choose the gradient boosting framework LightGBM [30], which is a tree-based learning algorithm, to train multiple models (the idea of stacking is described in detail in [24]). The data set is split into 120 million possible engagements for training and a further 24 million for validation and testing.

The data set provides user, tweet and engagement features, which are listed in Section A.1. In addition, they design several features. To analyse the semantic of the tweets they utilised the modified output of the pretrained multilingual BERT[18]. Furthermore, they extracted information about user profiles, such as the account age or the mainly used language. Among various other features, they also introduced a social-based group of features they refer to as graph features. These graph features exploit social structures from the network. Since relationships between users are useful to predict user engagements, they extract the first and second-degree connection from a social follow graph, which they construct from the data set. Furthermore, they employ PR to represent users' social influence in the network. In addition, they construct a second graph which is based on the engagement type like. Nodes represent users and the edges represent like engagements. They utilise a simple version of a personalised PR: a random walk restarts from the engaging user while counting the number of visits to the user engaged with. They assume that users who have common preferences and favourite topics are more likely to engage with each other.

The model is evaluated using two types of metrics, Precision-Recall Area Under Curve (PR-AUC) and Relative Cross Entropy (RCE). RCE indicates the improvement of a model prediction compared to the naive prediction – a prediction that outputs the engagement rate of the training set without considering the tweet and user features. RCE is computed as follows:

$$RCE = \frac{(CE_{naive} - CE) \cdot 100}{CE_{naive}} \tag{3.1}$$

with CE defined as the cross entropy of the prediction and CE_{naive} as the cross entropy of the naive prediction. PR-AUC indicates a label imbalance in the data set (additional information on PR-AUC can be found in [14]).

In summary, Goda, Agata, and Matsumura develop a RS that uses features, which are generated by PR, among others.

3.4 FAIRNESS-AWARE LINK ANALYSIS

In 2020, Tsioutsiouliklis et al. published a novel idea to include fairness in link analysis and in particular for PR [50]. They provide a parity-based definition of fairness, where the nodes of the network belong to two groups based on specific characteristics (for instance gender or demographics). Thus, the algorithm must allocate a proportion of PR to the members of each group. In their investigation, the graph $G = (V, E)$ with order n has unweighted edges and the set of users is divided into two groups – protected (i.e. under-represented) and unprotected. In this context, they define a fairness policy as follows:

> "A link analysis algorithm is ϕ-fair, if the fraction of the total weight allocated to the members of the protected group is ϕ. The value of ϕ is a parameter that can be used to implement different fairness policies."

They propose two approaches with the difference of enforcing fairness locally and globally. The Locally Fair PageRank (LFPR) imposes a fair allocation for each node and thus adjusts the transition matrix, which corresponds to a "fair random walk". More specifically, each individual node acts fairly by allocating a fraction ϕ of its PR to its protected neighbours and the remaining fraction $1 - \phi$ to its unprotected neighbours. However, there are nodes that only link to nodes belonging to one group. In this case, the affected fraction is distributed to all nodes in the graph from the particular group. Let P be the set of protected nodes and $out_P(i)$ be the number of outgoing edges from i to protected nodes. They define the stochastic transition matrix T_P that handles transitions to protected groups, or creates new jumps to protected nodes if such links do not exist:

$$T_P[i,j] = \begin{cases} \frac{1}{out_P(i})) & \text{if } (i,j) \in E \text{ and } j \in P \\ \frac{1}{|P|} & \text{if } out_P(i) = 0 \text{ and } j \in P \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

The transition matrix T_U for the unprotected nodes U is formulated similarly. As a result, the transition matrix $\tilde{T}$ of LFPR is defined as:

$$\tilde{T} = \phi T_P + (1-\phi)T_U \tag{3.3}$$

The equation of PR, which we derived in Section 2.3.3, remains the same except the modified transition matrix:

$$p^T = \alpha p^T \tilde{T} + (1-\alpha)v^T \tag{3.4}$$

Tsioutsiouliklis et al. prove that this microscopic view of fairness, where each node acts independently of the other nodes in the graph, results in a macroscopic view of fairness.

In comparison to the local concept presented above, the Fairness-Sensitive PageRank (FSPR) has a global approach. FSPR keeps the transition matrix T fixed but changes the jump vector v in order to comply with the fairness policy. The PR scoring vector can be written as a linear function of the jump vector v:

$$p^T = (1-\alpha)v^T[\mathbb{1} - \alpha T]^{-1} \tag{3.5}$$

Let $p[P]$ denote the PR mass that is allocated to the nodes of the protected group. For the FSPR to be fair, one has to find a jump vector v that satisfies $p[P] = \phi$.

Moreover, they consider the task of achieving fairness while minimising the utility loss $L(p_{FA}, p_O)$, understood as the euclidean distance of the output p_{FA} and the output p_O of the original unconstrained PR:

$$L(p_{FA}, p_O) = ||p_{FA} - p_O||_2 \tag{3.6}$$

Tsioutsiouliklis et al. conduct experiments with different data sets to investigate the conditions under which PR unfairness emerges and evaluate the utility loss in enforcing fairness. Besides two data sets that present networks of books and blogs about US politics, they study the author collaboration network DBLP[1] and a political retweet

1 https://dblp.org

graph from Twitter. The latter graph consists of $61,157$ edges and $18,470$ users of which 61% are assigned with the protected attribute of being political left.

The research of Tsioutsiouliklis et al. offers a new concept to include fairness in PR. The usage of node attributes introduces new approaches to adjusting PR according to a defined policy. We will further refer to this research and state points and problems why FSPR and LFPR are not successful or practicable for our concrete problem and the given data set.

4 PROPOSED METHOD AND IMPLEMENTATION

In this chapter we will present the method and implementation of FAPR and prove the concept using a data set from the online social network Twitter. Section 4.1 first outlines how Twitter works and then introduces the data set that was provided as part of the RecSys Challenge 2021. Then, we will turn our attention to the practical task. Section 4.2 describes the preprocessing of the data including the graph structure and the sampling. In Section 4.3, we will present our method to create a fairness-aware transition matrix that enhances fairness locally.

The goal of this research is to investigate the effectiveness of popularity debiasing strategies for a RS. However, within the scope of this work, it would not be feasible to implement a full-scale RS that is accurate but also enables an in-depth analysis of fairness. Thus, we will concentrate on PR, an algorithm that generates social-based features a model-based RS refers to, among other features. Although a RS includes further features, the usage of PR in particular illustrates how a potentially biased PR can influence a RS. If the features are biased and the model uses this information, the RS tends to be biased too. Therefore, we analyse and modify the PR algorithm to reduce the popularity bias. The idea of adjusting an algorithm depending on protected attributes, in this case, the popularity, can be applied to other similar concepts that generate features.

As introduced in Section 3.4, Tsioutsiouliklis et al. proposed a local method that allocates PR weights according to a defined policy supporting a protected group. However, this work is restricted to a graph with unweighted edges and a bipartite group classification: protected and unprotected. Thus, we introduce an algorithm that handles weighted and also large networks. Moreover, our proposed implementation can handle different fairness policies with an arbitrary number of group classifications.

4.1 DATA SET

Since our goal is to examine and reduce the effect of the popularity bias on PR, and thus indirectly on a RS, we require real-world data on which a RS would normally rely. As part of the RecSys Challenge 2021, Twitter released data for participants to implement a RS predicting tweet engagements. The size of the data set as well as the available features provide a suitable setting in order to investigate fairness in a RS.

In order to understand the data set and the graph, which we construct based on Twitter's social network, we will first introduce the setting of Twitter. Then, we will explain the setting and the goals of the RecSys Challenge, which influence our definition of fairness. Finally, we will outline how we formulate our research objectives in relation to the RecSys Challenge.

4.1.1 *Online Social Network Twitter*

The social network Twitter is a web-based micro-blogging service, which was first launched by Obvious in 2006 [27]. Twitter is a platform on which users publish and engage with short posts known as *tweets*. A tweet is limited to 140 characters and is publicly visible by default meaning that the post appears on the *public timeline*. However, users are able to restrict the message delivery that only those who have subscribed to the user's feed have access to the tweets. In this case, users *follow* other users – including individuals as well as organisations or companies. Since unilateral following is possible, users can individually decide which users they want to follow according to their preferences. This explicit feedback could be exploited by RS to recommend the content of the followed users.

In addition to the text, Twitter offers further possibilities to edit and tailor tweets. Users can add URL links or use hashtags, which are words or phrases prefixed with a "#" sign, to tag and cross-reference content. Furthermore, they can attach media, such as photos, audio or videos.

Users that are registered can publish tweets as well as engage with them by choosing from three options. Besides liking, one can also comment on the tweet, which is then displayed alongside the tweet. The last option describes the process of a *retweet* where users forward a post to their own feed. When users repost a tweet, they get the additional option to add a comment to their retweet. The usual use of Twitter can thus be summarised as follows: When you are not posting tweets to express yourself, you can scroll through your timeline to see content preferentially from users you are following. Then, you can engage with these tweets.

Twitter, with its over 199 million daily active users, is making an important contribution to the fact that, apart from the few countries where Twitter is censored, the world is connected by constantly exchanging information.

4.1.2 *RecSys Challenge 2021*

The Association for Computing Machinery organises an annual conference on Recommender Systems (RecSys) along with a challenge.

As part of the RecSys Challenge 2021 [1], Twitter released data in order for participants to predict the probability of different engagement types of a target user while providing fair recommendations.

The public data set of close to 1 billion data points is split into 75% for training and 25% for evaluation and testing. Each data point contains the tweet along with engagement features, user features, and tweet features. Section A.1 describes the 24 features of a data point. In summary, a data point describes the situation in which a user sees a tweet in the timeline and possibly engages with it. If a user does not interact with the displayed tweet, the data point is called pseudo-negative, reflecting implicit negative feedback. Furthermore, a distinction is made between four different types of engagement. In addition to the three major options mentioned above – like, retweet and comment – there is also a feature indicating a retweet with a comment. A data point can consist of a tweet interaction with multiple engagement types. For instance, a user could retweet and like a tweet, which is represented by a single data point. Table 4.1 provides statistics of the training data set.

	#	%
Data points	747,694,282	
Pseudo-negatives	37,5757,100	50.26%
Like	297,005,146	76.27%
Retweet	65,440,419	16.81%
Reply	21,708,921	5.57%
Retweet with comment	5,243,892	1.35%
Languages	66	

Table 4.1: Description of the training data set

The accuracy of the predictions is measured by the two metrics RCE (see Equation 3.1) and Average Precision (AP) (additional information on AP can be found in [14]), which compare the predicted and the actual engagement from the test data set.

Since fairness is a societal concept with a wide range of interpretations, the initiators of the challenge have decided to use a popularity-based metric. The quality of the recommendations should not depend on the popularity of the author that created the tweet. Therefore, they divide the authors into five groups according to their popularity (computed as quantiles of the authors' number of followers in the test set). Then, both metrics AP and RCE are computed for each group. The final score is the average of the scoring across each group.

1 https://recsys.acm.org

Twitter had also made data available for the RecSys Challenge in 2020. The task was the same apart from the aspect of fairness, which was introduced in 2021. In addition to that, the amount of data including the test set increased from 200 million to 1 billion data points.

We alienate ourselves from the actual task of the challenge and look more closely at how to incorporate fairness into RSs. Since this work examines fairness in terms of the popularity bias, we adopt the concept of the popularity-based metric.

4.2 DATA PREPROCESSING

This section describes the preprocessing steps necessary to extract a subgraph for the analysis and modification of PR. Since the personalised ranking from PR mainly retrieves information from the neighbourhood of a node, we do not bias our investigation if we focus on a subgraph of the network. Additionally, memory limitations play a role, otherwise, we could not efficiently adjust the algorithm in terms of the fairness parameter.

4.2.1 *Graph-Based Representation of Twitter's Network*

As PR measures the influence of a node in a network, we need to transform Twitter's social structures into a graph $G = (V, E)$. The provided features (see Section A.1) of each data point suggest constructing an engagement graph with weighted edges. Another possibility would be to create an unweighted graph in which the edges would describe the following status. However, we would lose information if a user had interacted with another user more than once. Thus, we define our engagement graph as $G = (\{\text{User ID}\}, E)$, with edges $w_{ij} = e$, if user i has engaged e times with user j. In other words, a node represents a Twitter user, identified by the unique user ID. A directed and weighted edge provides information about how often a user has engaged with the available tweets of another user. Furthermore, we assign the number of followers of each user as a node attribute. The concept of the engagement graph is illustrated by Figure 4.1.

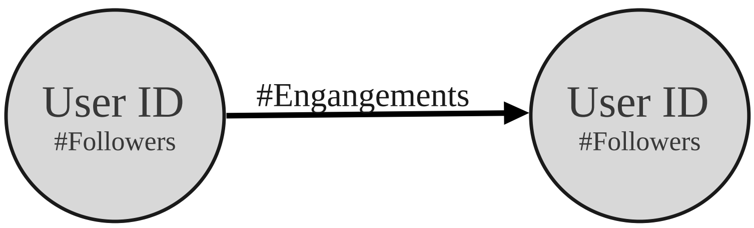

Figure 4.1: Engagement graph: users are connected through engagements

Although one data point distinguishes between four different types of engagement, we will only filter for like and retweet. Both engagement types generally indicate positive feedback for the particular tweet, as opposed to commenting or retweeting with a comment. The latter two interactions do not inevitably reflect a positive reaction and therefore should preferably be analysed, e.g. with a content-based approach, to generate a diverse classification of the engagement behaviour. However, the objective of this work is not the content of the tweets, but on the social structures and since 94.30% of all engagements are covered by like or retweet, we only focus on the positive feedback. In addition, we filter by the most commonly used language to ensure that the sample graph describes a more connected network. Twitter hashed the user IDs as a string with 16 characters. To improve the complexity of search algorithms, we additionally map the user IDs as an integer.

Even when applying the three filters - most common language and, like or retweet - $19,219,528$ users remain. The resulting subset of engagements does not directly indicate to which degree these users are connected. However, the connectivity of the subgraph is important since we investigate PR with a local approach that integrates fairness by the transition probability and thus by the node's neighbourhood. To extract the largest connected subgraph, we will traverse the filtered engagements using BFS with the most connected user as the starting seed S (see Listing 2.1). As $124,624$ users engaged with user 2090 resulting in $197,177$ engagements, we choose user 2090 as the starting seed S. Since we have stored the engagements in an adjacency list where a key represents a user and the values represent the users who engaged with the key-user, the BFS is based on ingoing edges and thus the graph is constructed "reversed". Figure 4.2 illus-

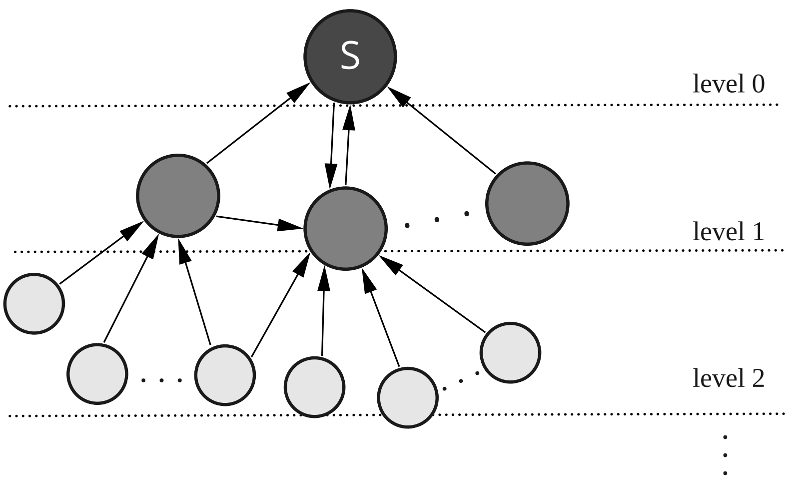

Figure 4.2: Starting a BFS at node S to construct the engagement graph

trates the procedure. If user S interacted with another user from the network, each node would have at least one outgoing edge and thus there would be no dangling nodes[2].

After starting BFS at seed S, the constructed graph contains $12,176,671$ nodes and $40,897,206$ edges. In comparison, Tsioutsiouliklis et al. use in their investigation of FAPR a Twitter graph with $18,470$ users. The number of nodes is too high for a detailed investigation and also not necessary, as the personalised FAPR will not traverse the entire graph. Thus, we need to further reduce the order of the graph and draw a sample.

4.2.2 *Sampling*

We use the above final network with $12,176,671$ users as a baseline to draw a sample having the same distribution. More precisely, we want our sample to keep the same probability distribution of the user popularity (e.g. the number of followers). With conventional sampling methods, one could select data points such that the sample iteratively meets the distribution conditions. However, this approach would destroy the graph structure if we selected users from the data set without considering their connection and edges. Thus, we first need to generate subgraphs. These can afterwards be checked to see if they match our distribution.

The goal of the personalised PR is to identify user similarities and possible relevant users from the network. Stanley Milgram's "small world study" [49] proposed the idea that all people, in general, can be connected via an average number of six steps. Zhang and Tu also confirm this hypothesis for online societies [53]. As the study shows, a social network has the property that users are ideally connected to many other users via short-length paths. This means that possibly relevant users can already be found in the immediate vicinity of a user. Therefore, it is reasonable to create an induced subgraph based on nodes from a chosen neighbourhood.

In order to find a more connected subgraph, we focus on finding a wide neighbourhood of node S. Therefore, we determine all nodes i with $dist(i, S) \leq k$. In other words, we find a neighbourhood that is at most k steps away from S. For $k = 20$, we get a graph with $3,697,785$ nodes and $7,254,686$ edges. The average degree of the nodes in this subgraph is 1.96 compared to 3.36 of the whole graph. One reason for this is that we cut out the subgraph and thus leave out the edges at the boundary of the defined neighbourhood.

Figure 4.3 illustrates the popularity distribution of this sample compared to the supergraph. The largest difference of 9.95 percentage points occurs in the first bin that reaches from 0 to 120 followers. The remaining bins show an average difference of 0.0203 percentage

2 The graph we will construct does not have any dangling nodes.

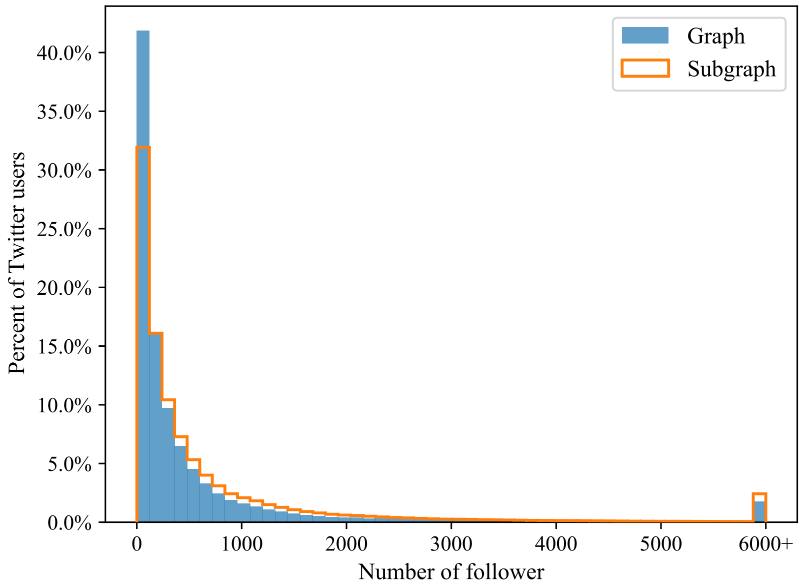

Figure 4.3: Follower distribution of the sample in comparison to the baseline

points. The sample meets our requirements sufficiently and shows the same pattern of popularity distribution. Consequently, we use the sample for the further investigations.

Since we are interested in a personalised PR, the additional question arises as to which node we take as a seed, at which the random walk of PR restarts. In order to observe the effect of the popular user S in an unbiased setting, we choose a user u that is the furthest away, i.e. with $dist(u, S) = 20$. Table 4.2 shows a selection of candidate seeds. User 1474069 with 362 followers interacts a total of 23 times

USER ID	#OUTGOING EDGES	TOTAL WEIGHT OF OUTGOING EDGES	#FOLLOWER
1474069	**9**	**23**	**362**
35085	21	139	456
344654	3	3	218
880588	3	11	284
519819	2	3	1500

Table 4.2: Possible seeds for personalised PageRank

with 9 different users. This represents a scenario we can expect from an ordinary Twitter user who provides enough information to receive personalised content.

In conclusion, we will apply FAPR personalised on user 1474069 in the subgraph that contains $3,697,785$ nodes.

4.3 FAIRNESS-AWARE PAGERANK

In this section, we will first introduce an open-source PR, which will serve as the basis for implementing a FAPR. We will conclude this section by addressing RQ2: *How can we adjust the PageRank algorithm to reduce the adverse effect of user popularity?*

Since our goal aims to reduce the popularity bias and thus create a fairer representation of all nodes, we will work with a local fairness approach. Every node should benefit from the modifications and be presented fairly. As a result, we will compute the transition matrix in terms of a fairness policy ϕ_i.

4.3.1 *NetworkX - PageRank*

NetworkX[3] is a Python package for network analysis including a graph implementation and tools for the study of the structure and dynamics of networks. Among other open-source algorithms for link analysis, three PR approaches are provided which utilise different mathematical libraries. One PR algorithm uses NumPy's LAPACK[4] package for computing eigenvectors, which works the fastest and most accurate for small graphs. However, for our dimension, it does not converge at all. Hence, we use the method based on SciPy, which takes advantage of the sparsity of the adjacency matrix.

The method takes as an input a NetworkX graph G and several optional parameters, which are listed in Table 4.3 The method returns a

PARAMETER	TYPE	FUNCTION
α	float	Damping factor; default 0.85
personalisation	dict	Personalisation vector with value for each node
weight	key	Edge data key to use as weight; if None weights are set to 1
dangling	dict	Creating outedges for dangling nodes; default is the personalisation vector (uniform if not specified)

Table 4.3: Parameters for NetworkX's `pagerank_scipy`

dictionary of the nodes in G with PR as the value. At the beginning, the algorithm calls `nx.to_scipy_sparse_matrix` which returns the transition matrix of the graph G. This is the part where we instead use our novel method to create a fairness-aware transition matrix. The rest of the method remains unchanged. Based on the equation of PR

3 `https://networkx.org`
4 `http://www.netlib.org/lapack/#_presentation`

that we derived earlier (see Equation 2.13), the algorithm iteratively updates the PR scoring vector up to a certain error tolerance.

4.3.2 *Fairness-Aware Transition Matrix*

Given a weighted graph, we obtain a transition matrix T which describes the transition probabilities of reaching the neighbouring nodes within one step. Our goal is to adjust these probabilities so that the weights of PR are distributed according to a fairness policy. While explaining the equations we will demonstrate and visualise our concept using an example where we divide the user set into $k = 5$ equal-sized groups. The nodes attributes are defined as $i \in \{1, 2, 3, 4, 5\}$ that describes the group a node belongs to. Figure 4.4 illustrates the outgoing edges of a node u (with an arbitrary group classification i) labelled with the corresponding weights w_{uv} and the unprocessed transition probabilities t_{uv}. In this example, the probability of a random walk

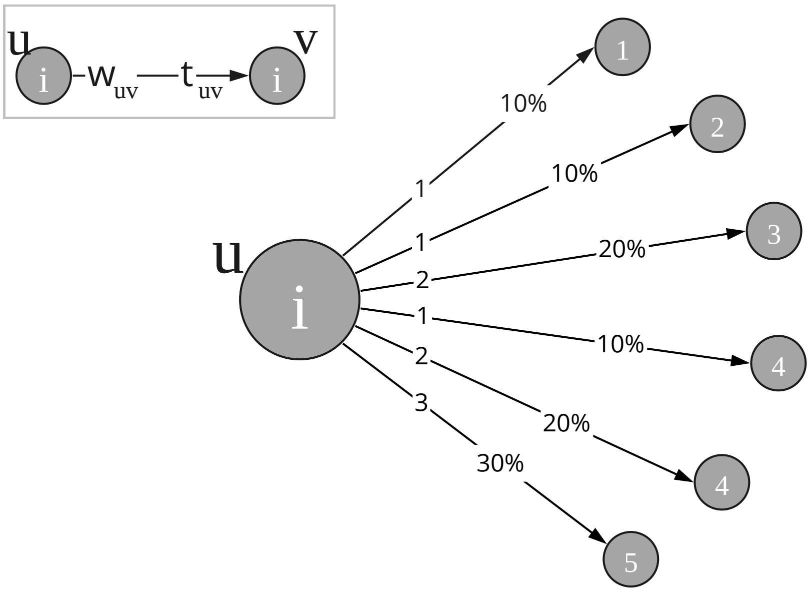

Figure 4.4: Outgoing edges with weights w_{uv} and transition probabilities t_{uv}

arriving at nodes of the group 4 or 5 is 60%, although both groups, in general, make up only 40% of all nodes due to the group classification. If this ratio is also found for many other nodes, the graph is affected by a bias based on the node attribute. Thus, we consider this particular distribution as not fair we want to adjust the transition probabilities of each group V_i so that they target the fairness parameters ϕ_i. Since the outgoing degree of a node can vary, we have to adjust the calculations resulting in two different equations. Therefore, we first look at Figure 4.4, which describes the "perfect" scenario - a node u points to nodes v of each group V_i. In this case, FAPR can already achieve the fairness policy through the neighbours. The sparsity of the transition matrix remains the same and the fairness policy is not only targeted as a value but completely realised. In order to include the fairness policy, we iterate through each group and compute the modified transition probabilities $\tilde{t}_{uv}$ so that they match the fairness parameters:

$$\phi_i = \sum_{v \in V_i} \tilde{t}_{uv} \tag{4.1}$$

For illustration, we set the uniformly distributed fairness parameters $\phi_i = 0.2$ for $i \in (1, 2, 3, 4, 5)$ leading to modifications, which are shown in Figure 4.5.

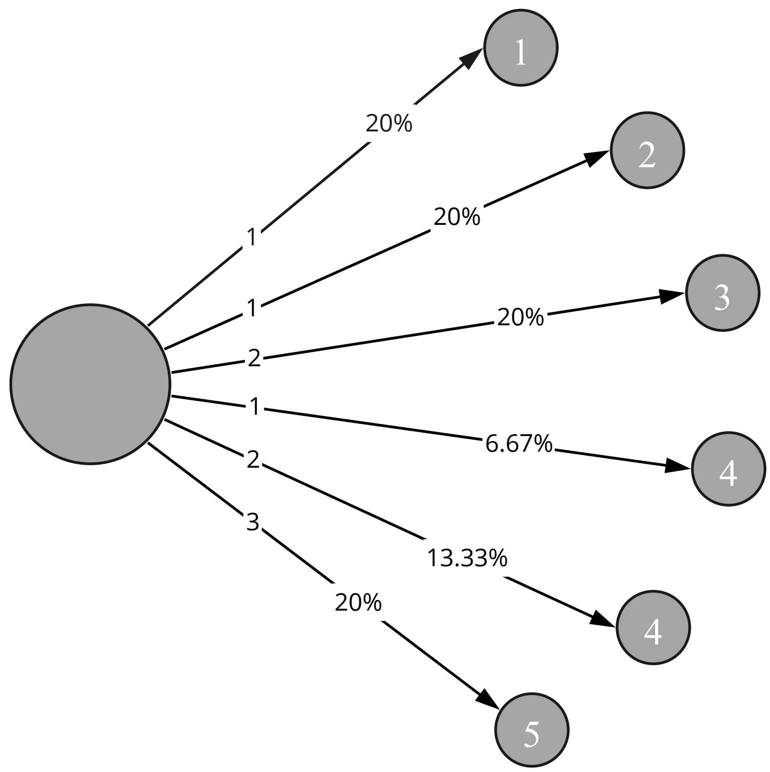

Figure 4.5: Modification: The node is connected to nodes of all groups. The edge is labelled with w_{uv} and the modified transition probabilities $\tilde{t}_{uv}$.

However, not every node necessarily has a neighbour of each group V_i resulting in an "unperfect" scenario. In order to still meet the fairness policy, ϕ_i must be distributed in a different way. Tsioutsiouliklis et al. propose to distribute the remaining part ϕ_i to all nodes $v \in V_i$. This approach does not work for large graphs, as the values become extremely small and the sparsity of matrix T gets lost. Therefore, we utilise a novel method that considers $K_2(u)$, the second-degree neighbourhood of a node u, i.e. the nodes k with $dist(u,k) = 2$. We distribute the part ϕ_i to all nodes $k \in V_i$ belonging to the neighbourhood according to the weights of the entire path:

$$\phi_i = \sum_{k \in V_i \cap K_2(u)} \tilde{t}_{uk} \tag{4.2}$$

We are interested in a formula to compute the a new transition probability t_{uk}. Let (u, v, k) be one or more paths $u \to v \to k$ between the fixed nodes u and k with $k \in V_i \cap K_2(u)$. The transition probability is computed as:

$$\tilde{t}_{uk} = \frac{\sum\limits_{\forall v:(u,v,k) \in E} w_{uv} + w_{vk}}{\sum\limits_{\forall l \in V_i \cap K_2(u)} \sum\limits_{\forall v:(u,v,l) \in E} w_{uv} + w_{vl}} \phi_i \tag{4.3}$$

Figure 4.6 describes the "unperfect" scenario and resulting modifications when nodes of group 2 only exist in the second degree neighbourhood. We artificially add transition probabilities to the nodes that

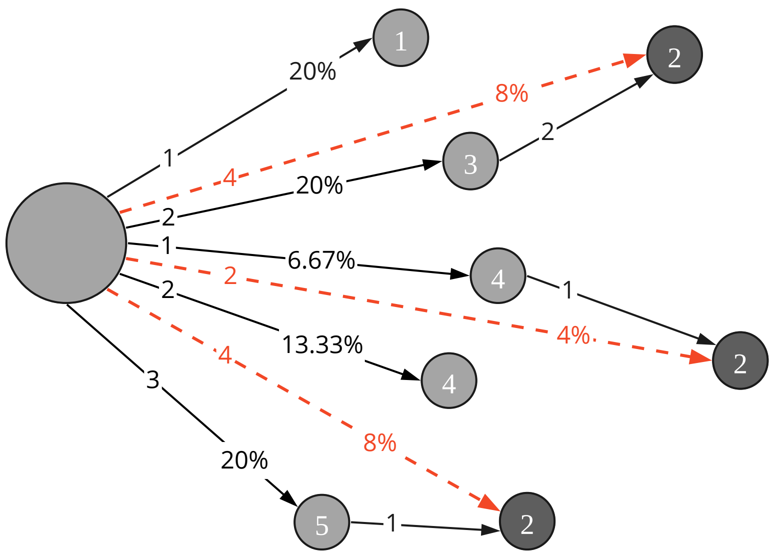

Figure 4.6: Modification: Transition probabilities to nodes of group 2 are added. The probability is calculated based on the weights of the paths.

are not directly connected. To clarify the computation, we review the lowest path traversing the nodes of groups 5 and 2. The weight of the path is $3 + 1 = 4$. Furthermore, the sum of the three paths together is $4 + 2 + 4 = 10$. Since the fairness ratio $\phi_2 = 20\%$ has to be assigned, the transition probability yields $\frac{4}{10} \cdot 20\% = 8\%$ for the lowest node from group 2.

If there are no nodes of a group in the surrounding area either, we discard this part of the weight. We do not want to add completely artificially constructed transition probabilities and thereby manipulate the structure of the graph. Consequently, the fairness policy is thus to be understood as a benchmark.

There is another scenario that needs to be kept in mind. Dangling nodes have no outgoing edge, which generally causes a problem for PR. Regardless of the fairness-aware approach, the personalised vector would serve as the transition probability for the dangling node.

As a result, we have developed a method that returns a fairness-aware transition matrix, which targets defined fairness parameters. In particular, the edges of the graph can be weighted and both the number of assigned user groups k as well as the fairness parameters ϕ_i are adjustable.

5 RESULTS AND DISCUSSION

This chapter presents the experimental results of applying FAPR on the social network of Twitter. First, we specify the popularity-based group classification we choose for the particular data set. Subsequently, we elaborate on the effect of FAPR on fairness in terms of the GGI. Finally, we address research question RQ3: *How do we optimise the trade-off between PageRank's utility and fairness?* Therefore, we define a utility loss function to examine various combinations of fairness parameters.

5.1 GROUP CLASSIFICATION

The sample, we generate in Section 4.2.2, consists of $n = 3,697,785$ nodes. FAPR requires a fairness policy ϕ_i for $i \in (1, ..., k)$, therefore the parameters ϕ_i and the number of groups k have to be chosen. Our goal is to classify the user into groups regarding their popularity, determined by their number of followers. Due to the metric property of this definition of popularity, we have many possibilities to choose both the number of groups k as well as the respective group sizes $|V_i|$. As the RecSys Challenge incorporates fairness into the accuracy metric by using 5 user groups of equal size, we set $k = 5$ and pass $\{1, 2, 3, 4, 5\}$ as the node attributes.

In order to classify each node depending on their popularity, we utilise a quantile-based discretisation function. Table 5.1 shows the resulting quantiles with the corresponding number of followers. We

GROUP	QUANTILE	#FOLLOWERS
1	$[0\% - 20\%]$	$[0, 60.0]$
2	$(20\% - 40\%]$	$(60.0, 173.0]$
3	$(40\% - 60\%]$	$(173.0, 382.0]$
4	$(60\% - 80\%]$	$(382.0, 929.0]$
5	$(80\% - 100\%]$	$(929.0, 46507085.0]$

Table 5.1: Quantile-based discretization of users in terms of their popularity

set the popularity group as a node attribute allowing FAPR to distribute the weights according to the defined fairness policy.

5.2 FAIRNESS

In order to quantify the resulting enhancement of fairness, we compute the GGI – an index that measures the disparity among the values of a frequency distribution (see Section 2.2.4) – of the original PR, based on the implementation of NetworkX, compared to FAPR. In our analysis, the Lorenz curve depicts popularity on the x-axis in relation to the PR on the y-axis. Therefore, the users are sorted in ascending order according to their number of followers and the corresponding PR scores are accumulated.

For the subsequent investigation of fairness, we will define a uniformly distributed fairness policy $\phi_i = 0.2 \quad \text{for } i \in \{1, 2, 3, 4, 5\}$. Both PR algorithms are personalised on the same node and the damping factor is set to default, i.e. $\alpha = 0.85$. Since the Lorenz curve is a discrete function, the range of the number of followers has to be binned in order to calculate the required area. We use 5 bins for the first histogram and subsequently 100 bins for the second histogram. Figure 5.1 shows the Lorenz curves of the original, unrestricted PR and FAPR. The line of equality describes the uniform distribution and serves to calculate the GGI. The last bin of the Lorenz curve of the original PR indicates the popularity bias: 20% of the most popular user obtain 70.55% of the PR. The GGI for PR is 0.4002 in comparison to the GGI of 0.1414 for FAPR. As a result, the difference of GGI is 0.2588 in absolute terms, yielding a relative value of 64.67%. Thus, a uniformly distributed fairness policy for FAPR results in a fairer allocation of PR.

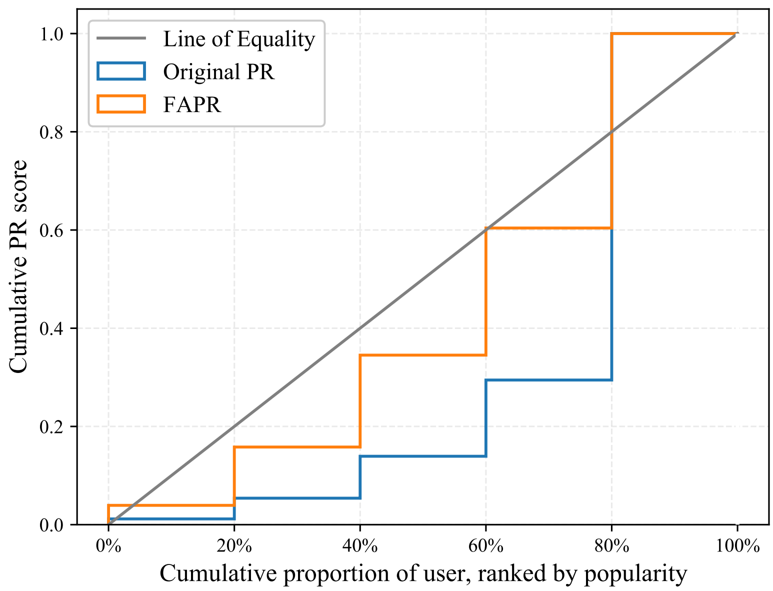

Figure 5.1: Lorenz curve for the orignal PR and FAPR using 5 bins

To derive more detailed insights we also set the bin size to 100. Figure 5.2 presents the same parameter choices, but the values are now binned in percentiles instead of quantiles. The most popular percentile of users gets 26.71% of PR for the original implementation and 13.82% for FAPR. Moreover, the histogram illustrates the impact of the quantile-based fairness policy, as the approximated Lorenz curve shows slight differences in slope at the quantile boundaries. Furthermore, the popularity bias leads to an exponential trend that can be seen in the quantiles.

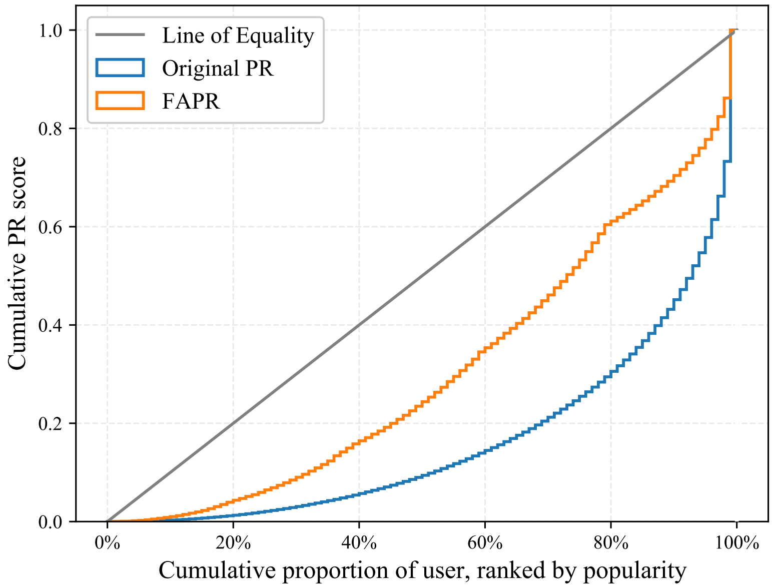

Figure 5.2: Lorenz curve for the orignal PR and FAPR using 100 bins

USER ID	#FOLLOWER	i	PR_O	PR_{FA}	$PR_O - PR_{FA}$
2090	46507085	5	0.008718	0.002844	0.005874
3846	454097	5	0.004539	0.000963	0.003576
1859	433940	5	0.004389	0.001167	0.003222
168469	150	2	3.365367e-06	0.001082	-0.001079
165416	368	3	3.676819e-06	0.001019	-0.001016
111227	428	4	3.563828e-06	0.000882	-0.000878

Table 5.2: Uniformly distributed FAPR: Users with the largest impact after applying FAPR

Table 5.2 shows a selection of users receiving the greatest impact after applying a uniformly distributed FAPR. Consistent with the algorithm's goal, users of the most popular group 5 lose the most PR.

Although user 2090 suffers the greatest loss of PR, the user is the most influential in the network in both algorithms. The fact that both users 3846 and 1859 from the same group change their relative position in the PR ranking shows that FAPR takes the graph structure into account and that the popularity bias varies across neighbourhoods. Furthermore, the table underlines the desired goal that users with a smaller audience gain importance in PR. After the debiasing process, user 168469 that receives 44 engagements from 20 users is deemed more relevant.

5.3 COMPARISON FAIRNESS AND UTILITY

After having studied the fairness improvement of FAPR in comparison to the original, unrestricted PR, we will now optimise FAPR in terms of fairness and utility. After introducing the idea of utility, we will elaborate on the parameter selection process.

Our goal is to implement a fair PR but at the same time utilisable algorithm. For this purpose, we need to define what we regard as utilisable. The scoring vector of a modified PR should be able to map the properties of the specific network and to identify the relevant and influential users without significant data distortion. Since the original, unrestricted PR precisely accomplishes this task, we take it as a baseline to analyse the utility. In order to quantify the utility loss, we compute the RMSE of $p_{FA} \in \mathbb{R}^n$, the scoring vector of FAPR and $p_O \in \mathbb{R}^n$ from the original, unrestricted PR. Thus, we take into account every user of the network and compare their PR values for both implementations:

$$RMSE(p_{FA}, p_O) = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(PR_{FA}(i) - PR_O(i))^2} \tag{5.1}$$

We interpret the utility loss as a measure of the cost we have to pay to achieve fairness for the network. To minimise the utility loss, the optimal FAPR would remove uniformly the weights of the groups, which are over-represented, and subsequently add these weights uniformly to all nodes in the less represented groups. This results from the fact that the uniformly distributed vector has the smallest length among all distribution vectors [50].

The fairness parameters ϕ_i describe the targeted distribution of the PR for every user group i. For $k = 5$ groups, the parameters need to satisfy $\sum_{i=1}^{5} \phi_i = 1$. In order to adjust the parameters without varying the sum, we interpret the values of the distribution as a function $f : [1, 5] \to [0, 1]$ whose values in the domain yield:

$$\sum_{i=1}^{5} f(i) = 1 \tag{5.2}$$

The interpretation of a fair or equal algorithm suggests that a uniform distribution is a reasonable choice. Based on the above functional characteristics, the resulting uniformly distributed function is defined as $f(i) = 0.2$. To begin with, we will deal with linear functions. The linear functions are fixed at $f(3) = 0.2$ and tilted using different slopes. Figure 5.3 illustrates a selection of linear functions. We employ the functions to compute the fairness parameters for the 5 groups. The colour gradient, illustrated in Figure 5.4, characterises fairness: green indicates a fairer distribution supporting the underrepresented groups 1 and 2, while blue indicates a less fair distribution.

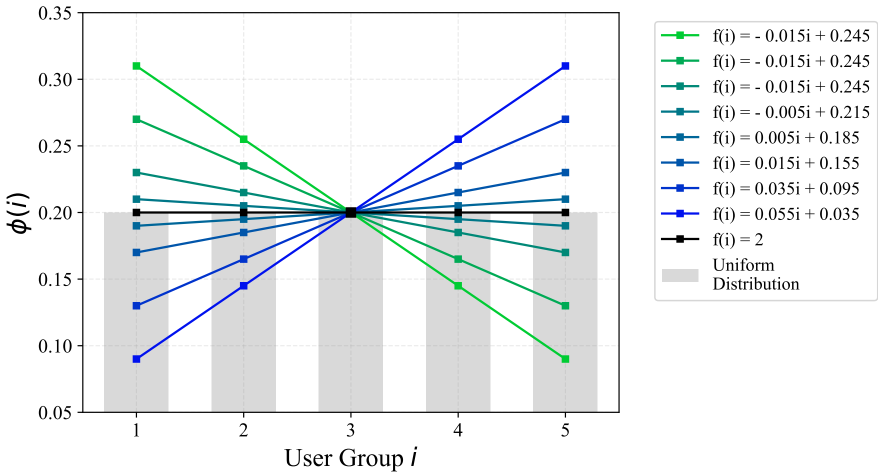

Figure 5.3: Parameters generated by linear functions

Figure 5.4: Colour gradient visualises fairness

After generating various parameters that have a linear relation, we investigate exponential functions. Since the Lorenz curve for 100 bins shows the exponential characteristic of popularity bias, we try to counteract this bias directly. Figure 5.5 describes exponential functions that satisfy Equation 5.2, using the same definition of the colour gradient. As the exponential functions behave linearly in the area of the constant function $f(i) = 2$, they are omitted. Compared to the linear function, these distributions are not symmetric and they are slightly shifted to the left, which means that $\phi(3)$ yields less than 0.2.

In the following, we compare the utility loss defined by RMSE with the fairness indicated by the GGI for multiple fairness policies gener-

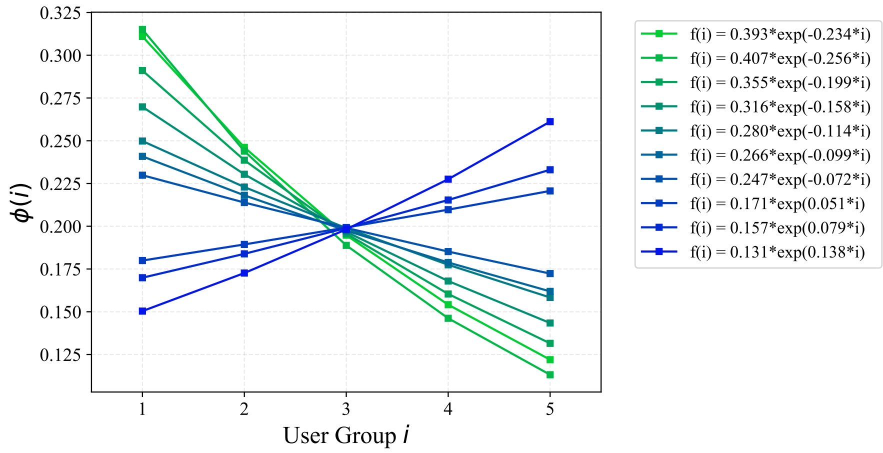

Figure 5.5: Parameters generated by exponential functions

ated above. The FAPR algorithm is personalised on node 1474069 (see Table 4.2) and the damping factor is set to default, i.e. $\alpha = 0.85$. Figure 5.6 plots the GGI of FAPR on the x-axis against the corresponding utility loss on the y-axis. In the graph, 10 fairness policies are based on exponential functions and 12 on linear functions. In addition, there are two points that represent a normal and uniform distribution of the parameters. The original PR obtains a GGI of 0.40 which is shown as a vertical line. For reference, the smaller the GGI, the fairer the algorithm.

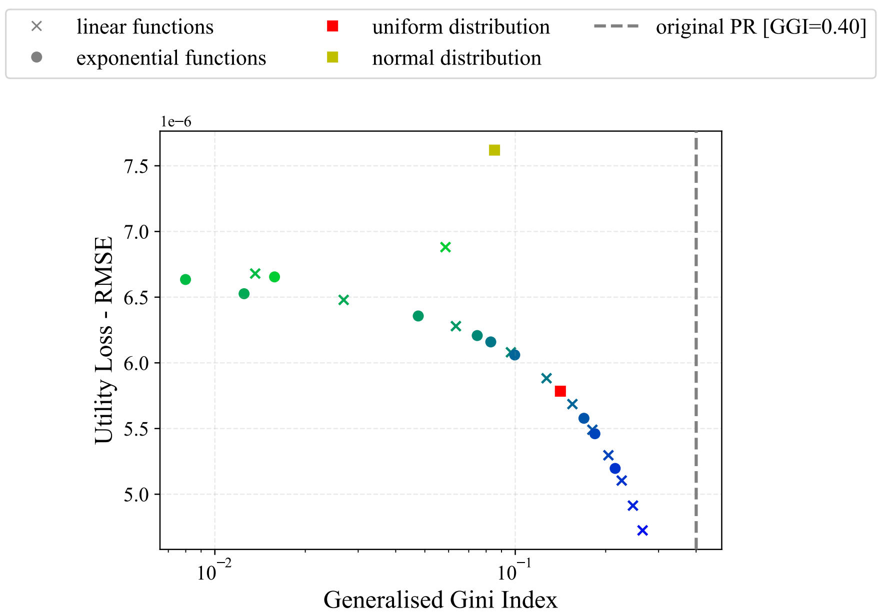

Figure 5.6: Utility loss [RMSE] compared to fairness [GGI]

As expected, the results indicate, that fairness is associated with a loss of utility. Moreover, the fairness policy must clearly counteract the bias of the data. The configuration which follows the normal distribution with a mean of $\mu = 3$ and a variance of $\sigma^2 = 0.8$ obtains the highest utility loss of $7.62 \cdot 10^{-6}$ in our test. To minimise the popularity bias, the unpopular groups rather than group 3 must be assigned more weight. Since some fairness policies overweight unpopular groups, the Lorenz curve lies above the line of equality, turning the GGI unfair again.

The results show that the utility loss is proportional to the GGI. Since both the linear and exponential distribution functions adequately address the problem of the popularity bias, a fairness policy with $k = 5$ groups shows no significant differences between the choice of both distributions. However, the normal distribution illustrates that the fairness policy of FAPR needs to be tailored and thus optimised according to the type of bias.

5.4 DISCUSSION

The above results of utilising FAPR clearly indicate that debiasing PR entails utility loss. The various examined fairness policies provide a guideline as to which parameters lead to which combination of utility loss and GGI. Since the GGI of the original PR is known, we are now able to choose among fairness policies to obtain a fairer PR. Among these fairer algorithms, we can compare the utility losses and decide on a policy depending on our desired outcome. The desired outcome on this data could, for instance, be to reduce the original GGI from 0.40 to 0.20. The results reveal that the exponential distribution $(0.15, 0.17, 0.195, 0.225, 0.26)$ with a utility loss of $5.20 \cdot 10^{-6}$ would achieve this goal. The utility loss should be interpreted as a measure that reflects the degree of manipulation of the network structure determined by the original PR. Since the fairness policies are integrated according to the same allocation concept, the utility loss relatively indicates which policies distort the network more. This relative measurement is sufficient to evaluate and benchmark the debiasing strategies in our research scope.

With regard to the optimisation problem, our evaluation does not conclude a concrete optimal parameter selection. However, they reveal that the fairness strategies need to be adapted to the characteristics of the underlying bias. Even though the fairer parameter choices based on linear and exponential functions counteract the popularity bias, we cannot extract an optimal solution due to the approximated linear relationship between fairness and utility loss. The linear relationship results from the fact that when a fairness parameter is modified, the modification is passed on locally to each node and is included in the linear calculation of the weight.

We presented a local fairness method that distributed the weights within a second-degree neighbourhood if a node is not directly connected to a particular group. Consequently, a part of the weights may not be allocated. One solution to this problem would be to extend the range to a neighbourhood of k-th degree and to distribute the weights more globally. Since we do not want to manipulate PR's task of extracting potential relevant users, and we do not want to decrease the efficiency of the algorithm by using a fuller matrix, we have chosen $k = 2$. However, further research could be done on the choice of parameter k and the resulting utility loss could be compared to our approach.

Furthermore, the algorithm could be optimised with regard to an online application. Instead of generating the matrix with all the nodes, one could consider more dynamic and more limited random walks. One could set limits beyond which a node is no longer classified as relevant for a particular user due to PR being too small.

6 CONCLUSION AND OUTLOOK

Any remedy for bias starts with awareness of its existence.

— *Ricardo Baeza-Yates*, 2018

In this work, we investigated the effectiveness of a popularity debiasing strategy. We presented a graph-based algorithm to identify and reduce the popularity bias in terms of a predefined fair policy.

We focused on the popularity bias, which frequently occurs in the process of generating recommendations. To identify the bias, we assigned the attribute popularity to the item set, in our case the users of a social network. After ranking the users according to their popularity, which we derived from the number of their followers, we adopted the idea of GGI. It describes the degree of equality of an allocation policy. Taking into account the popularity attribute, we can evaluate the equality of a given distribution and thus determine that popularity has an influence on the distribution in the corresponding item set. While this index originates from the analysis of income distributions, it precisely illustrates the popularity bias that arises in the distribution of PR.

In order to support unpopular users in PR, we implemented FAPR, an approach that locally mitigates the bias. Based on a fairness policy that assigns a popularity-based classification to nodes, the algorithm adjusts the transition probabilities to the neighbourhood. We included the number of engagements as the weight of edges in the computation and we also focused on adjusting a larger network compared to previous research for modifications of PR. Furthermore, our implementation offers the possibility to customise the fairness policy. The type of node attribute, the number of classified groups, and the corresponding group sizes can be changed individually. Thus, it is also possible to investigate and counteract further biases that are associated with other item attributes.

If we incorporate fairness and thus change the computation of PR, we lose utility, which we define by how FAPR's output differs based on the RMSE to the original unrestricted PR. We compared the trade-off between RMSE, representing utility loss, and GGI, representing fairness. The investigations with different fairness policies emphasise that the parameters must directly counteract the influence of the popularity bias in order to obtain optimal results with less utility loss. However, the studied parameter combinations, which are based on linear and exponential functions, have a linear relationship between

RMSE and GGI. Thus, the optimisation problem does not provide a unique solution and the parameters have to be selected depending on the desired fairness enhancement.

When designing and engineering a RS, it is important to take fairness issues into consideration. A first step in incorporating fairness into RS is to recognise unfair influences and build in measures to gain control over these influences. This is exactly the underlying aim of this work. We investigated how an algorithm can be changed based on established fairness policies and analysed how the outcome varies after the modifications. This work offers a versatile foundation and an approach to understanding how to handle the popularity bias.

The proposed method is an in-processing debiasing strategy for a model-based RS. However, a post-processing method would be more efficient as it would be applicable to all types of RSs. In addition, debiases strategies could be developed that go beyond popularity and take into account sensitive attributes such as gender or ethnicity, for example. With regard to OSNs, where most content is specifically related to individuals, versatile strategies would be a general social benefit. As web systems continue to grow in importance, RSs have an enormous impact on users. Therefore, research in fairness and especially debiasing strategies should and will continue to progress. However, we also believe that the users of a web system and in particular in an OSN should be involved and educated about the influence of RSs. For example, recommendations could be labelled or the origin of specific recommendations could be explained. In conclusion, research into the fairness of RS starts with awareness and ends with versatile implementations to gain influence over biases.

A
APPENDIX

A.1 FEATURES OF THE DATA SET PROVIDED BY THE RECSYS CHALLENGE

FEATURE	TYPE	DESCRIPTION
Text tokens	List[long]	Ordered list of Bert ids corresponding to Bert tokenization of Tweet text
Hashtags	List[string]	Tab separated list of hashtags (identifiers) present in the tweet
Tweet ID	String	Tweet identifier
Present media	List[string]	Tab separated list of media types. Media type can be in (Photo, Video, Gif)
Present links	List[string]	Tab separated list of links (identifiers) included in the Tweet
Present domains	List[string]	Tab separated list of domains included in the Tweet (twitter.com, dogs.com)
Tweet type	String	Tweet type, can be either Retweet, Quote, Reply, or Toplevel
Language	String	Identifier corresponding to the inferred language of the Tweet
Timestamp	Long	Unix timestamp, in sec of the creation time of the Tweet

Table A.1: Tweet Features

FEATURE	TYPE	DESCRIPTION
User id	String	User identifier
Follower count	Long	Number of followers of the user
Following count	Long	Number of accounts the user is following
Is verified?	Bool	Is the account verified?
Account creation time	Long	Unix timestamp, in seconds, of the creation time of the account

Table A.2: User features provided twice for both the engaging user as well as for the engaged with user

FEATURE	TYPE	DESCRIPTION
Engagee follows engager?	Bool	Does the account of the engaged tweet author follow the account that has made the engagement?
Reply Timestamp	Long	If there is at least one, unix timestamp, of one of the replies
Retweet Timestamp	Long	If there is one, unix timestamp, of the retweet of the tweet by the engaging user
Retweet with comment Timestamp	Long	If there is at least one, unix timestamp, of one of the retweet with comment of the tweet by the engaging user
Like Timestamp	Long	If there is one, unix timestamp, of the like

Table A.3: Engagement features

BIBLIOGRAPHY

[1] Himan Abdollahpouri. "Popularity Bias in Recommendation: A Multi-stakeholder Perspective". In: (2020). arXiv: 2008.08551. URL: http://arxiv.org/abs/2008.08551.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. "Controlling popularity bias in learning-to-rank recommendation". In: *RecSys 2017 - Proc. 11th ACM Conf. Recomm. Syst.* (2017), pp. 42–46. DOI: 10.1145/3109859.3109912.

[3] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. "Popularity-Aware Item Weighting for Long-Tail Recommendation". In: (2018). arXiv: 1802.05382. URL: http://arxiv.org/abs/1802.05382.

[4] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. "Managing popularity bias in recommender systems with personalized re-ranking". In: *Proc. 32nd Int. Florida Artif. Intell. Res. Soc. Conf. FLAIRS 2019* (2019), pp. 413–418. arXiv: 1901.07555.

[5] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. "The unfairness of popularity bias in recommendation". In: *CEUR Workshop Proc.* 2440 (2019). ISSN: 16130073. arXiv: 1907.13286.

[6] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. "Explore/Exploit Schemes for Web Content Optimization". In: Dec. 2009, pp. 1–10. DOI: 10.1109/ICDM.2009.52.

[7] Charu C. Aggarwal. *Recommender Systems: The Textbook*. 1st. Springer Publishing Company, Incorporated, 2016. ISBN: 3319296574.

[8] Saad Badaoui. "Black Lives Matter : A New Perspective from Twitter Data Mining". In: October (2020), p. 24.

[9] Ricardo Baeza-Yates. "Bias on the web". In: *Commun. ACM* 61.6 (2018), pp. 54–61. ISSN: 15577317. DOI: 10.1145/3209581.

[10] Zeynab Bahrami Bidoni, Roy George, and Khalil Shujaee. "A Generalization of the PageRank Algorithm". In: *ICDS 2014, Eighth Int. Conf. Digit. Soc.* c (2014), pp. 108–113.

[11] R. L. Basmann, K. J. Hayes, D. J. Slottje, and J. D. Johnson. "A general functional form for approximating the Lorenz curve". In: *J. Econom.* 43.1-2 (1990), pp. 77–90. ISSN: 03044076. DOI: 10.1016/0304-4076(90)90108-6.

[12] Lev Bogolubsky, Gleb Gusev, Andrei Raigorodskii, Aleksey Tikhonov, Maksim Zhukovskii, Pavel Dvurechensky, Alexander Gasnikov, and Yurii Nesterov. "Learning Supervised pagerank with gradient-based and gradient-free optimization methods". In: *Adv. Neural Inf. Process. Syst.* Nips 2016 (2016), pp. 4914–4922. ISSN: 10495258. arXiv: 1603.00717.

[13] Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. "Scaling Graph Neural Networks with Approximate PageRank". In: *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.* (2020), pp. 2464–2473. DOI: 10.1145/3394486.3403296. arXiv: 2007.01570.

[14] Kendrick Boyd, Kevin H. Eng, and C. David Page. "Erratum: Area under the Precision-Recall Curve: Point Estimates and Confidence Intervals". In: (2013), E1–E1. DOI: 10.1007/978-3-642-40994-3_55.

[15] Engin Bozdag. "Bias in algorithmic filtering and personalization". In: *Ethics Inf. Technol.* 15.3 (2013), pp. 209–227. ISSN: 13881957. DOI: 10.1007/s10676-013-9321-6.

[16] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. "Bias and Debias in Recommender System: A Survey and Future Directions". In: (2020), pp. 1–20. arXiv: 2010.03240. URL: http://arxiv.org/abs/2010.03240.

[17] Giovanni Luca Ciampaglia, Azadeh Nematzadeh, Filippo Menczer, and Alessandro Flammini. "How algorithmic popularity bias hinders or promotes quality". In: *Sci. Rep.* 8.1 (2018), pp. 1–7. ISSN: 20452322. DOI: 10.1038/s41598-018-34203-2. arXiv: 1707.00574. URL: http://dx.doi.org/10.1038/s41598-018-34203-2.

[18] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of deep bidirectional transformers for language understanding". In: *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.* 1.Mlm (2019), pp. 4171–4186. arXiv: 1810.04805.

[19] Mehrdad Dianati, Xeumin Sherman Shen, and Sagar Naik. "A New Fairness Index for Radio Resource Allocation in Wireless Networks". In: (2005), pp. 712–717.

[20] Minh-Phung T. Do, Dung V. Nguyen, and Loc Nguyen. "Model-based Approach for Collaborative Filtering". In: *Proc. 6th Int. Conf. Inf. Technol. Educ.* August 2010 (2010), pp. 217–225. URL: https://goo.gl/BHu7ge.

[21] Lars Eldén. "A Note on the Eigenvalues of the Google Matrix". In: (2004), pp. 1–3. arXiv: 0401177 [math]. URL: http://arxiv.org/abs/math/0401177.

[22] Facebook. "FB Earnings Presentation Q1 2021". In: *Facebook.Com* (2021).

[23] David Francis Gleich. "Models and Algorithms for Pagerank Sensitivity". AAI3382730. PhD thesis. Stanford, CA, USA, 2009. ISBN: 9781109444513.

[24] Shuhei Goda, Naomichi Agata, and Yuya Matsumura. "A Stacking Ensemble Model for Prediction of Multi-type Tweet Engagements". In: *ACM Int. Conf. Proceeding Ser.* (2020), pp. 6–10. DOI: 10.1145/3415959.3415994.

[25] Jonathan L. Gross, Jay Yellen, and Ping Zhang. *Handbook of Graph Theory, Second Edition*. 2nd. Chapman and; Hall/CRC, 2013. ISBN: 1439880182.

[26] Julia Heidemann, Mathias Klier, and Florian Probst. "Identifying key users in online social networks: A pagerank based approach". In: *ICIS 2010 Proc. - Thirty First Int. Conf. Inf. Syst.* (2010), pp. 1–21.

[27] Courtenay Honeycutt and Susan C. Herring. "Beyond microblogging: Conversation and collaboration via twitter". In: *Proc. 42nd Annu. Hawaii Int. Conf. Syst. Sci. HICSS* (2009). DOI: 10.1109/HICSS.2009.89.

[28] Ilse C.F. Ipsen and Teresa M. Selee. "Pagerank computation, with special attention to dangling nodes". In: *SIAM J. Matrix Anal. Appl.* 29.4 (2007), pp. 1281–1296. ISSN: 08954798. DOI: 10.1137/060664331.

[29] D. Jannach, L. Lerche, Iman Kamehkhosh, and Michael Jugovac. "What recommenders recommend: an analysis of recommendation biases and possible countermeasures". In: *User Modeling and User-Adapted Interaction* 25 (2015), pp. 427–491.

[30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie Yan Liu. "LightGBM: A highly efficient gradient boosting decision tree". In: *Adv. Neural Inf. Process. Syst.* 2017-December.Nips (2017), pp. 3147–3155. ISSN: 10495258.

[31] Jan Kratzer. *Encyclopedia of Social Network Analysis and Mining*. 2014. ISBN: 9781461461708. DOI: 10.1007/978-1-4614-6170-8.

[32] Amy N Langville and Carl D Meyer. "Deeper inside PageRank. Internet Mathematics". In: *Internet Math.* 1.3 (2004), pp. 335–380. URL: http://www.google.com/technology/index.html..

[33] Pan Li, Eli Chien, and Olgica Milenkovic. "Optimizing generalized pagerank methods for seed-expansion community detection". In: *Adv. Neural Inf. Process. Syst.* 32.NeurIPS (2019), pp. 1–12. ISSN: 10495258. arXiv: 1905.10881.

[34] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. "Feedback Loop and Bias Amplification in Recommender Systems". In: *Int. Conf. Inf. Knowl. Manag. Proc.* (2020), pp. 2145–2148. DOI: 10.1145/3340531.3412152. arXiv: 2007.13019.

[35] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. "A Survey on Bias and Fairness in Machine Learning". In: *ACM Comput. Surv.* 54.6 (2021), pp. 1–35. ISSN: 0360-0300. DOI: 10.1145/3457607. arXiv: 1908.09635.

[36] Gal Oestreicher-Singer and Arun Sundararajan. "Recommendation networks and the long tail of electronic commerce". In: *MIS Q. Manag. Inf. Syst.* 36.1 (2012), pp. 65–83. ISSN: 21629730. DOI: 10.2307/41410406.

[37] Lawrence Page and Sergey Brin. "The anatomy of a large-scale hypertextual Web search engine". In: *Comput. Networks* 30.1-7 (1998), pp. 107–117. ISSN: 13891286. DOI: 10.1016/s0169-7552(98)00110-x.

[38] Laura Patnaude, Carolina Vásquez Lomakina, Akshat Patel, and Gulhan Bizel. "Public Emotional Response on the Black Lives Matter Movement in the Summer of 2020 as Analyzed Through Twitter". In: *Int. J. Mark. Stud.* 13.1 (2021), p. 69. ISSN: 1918-719X. DOI: 10.5539/ijms.v13n1p69.

[39] Sigit Priyanta and I. Nyoman Prayana Trisna. "Social network analysis of twitter to identify issuer of topic using PageRank". In: *Int. J. Adv. Comput. Sci. Appl.* 10.1 (2019), pp. 107–111. ISSN: 21565570. DOI: 10.14569/IJACSA.2019.0100113.

[40] P. Ravi Kumar, K. L. Alex Goh, and K. S. Ashutosh. "Application of Markov chain in the PageRank algorithm". In: *Pertanika J. Sci. Technol.* 21.2 (2013), pp. 541–554. ISSN: 01287680.

[41] Santanu Saha Ray. *Graph theory with algorithms and its applications: in applied science and technology*. Springer Science & Business Media, 2012.

[42] Francesco Ricci, Lior Rokach, and Bracha Shapira, eds. *Recommender Systems Handbook*. Springer, 2015. ISBN: 978-1-4899-7636-9. DOI: 10.1007/978-1-4899-7637-6. URL: https://doi.org/10.1007/978-1-4899-7637-6.

[43] Arnon Rungsawang, Komthorn Puntumapon, and Bundit Manaskasemsak. "Un-biasing the link farm effect in PageRank computation". In: *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA* (2007), pp. 924–931. ISSN: 1550445X. DOI: 10.1109/AINA.2007.143.

[44] Daniel Schall. *Social Network-Based Recommender Systems*. 2015, pp. 1–126. ISBN: 9783319227351. DOI: 10.1007/978-3-319-22735-1.

[45] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai Cheng Yang, Alessandro Flammini, and Filippo Menczer. “The spread of low-credibility content by social bots”. In: *Nat. Commun.* 9.1 (2018). ISSN: 20411723. DOI: 10.1038/s41467-018-06930-7. arXiv: 1707.07592. URL: http://dx.doi.org/10.1038/s41467-018-06930-7.

[46] Ayan Sinha, David F. Gleich, and Karthik Ramani. “Deconvolving feedback loops in recommender systems”. In: *Adv. Neural Inf. Process. Syst.* June (2016), pp. 3251–3259. ISSN: 10495258. arXiv: 1703.01049.

[47] Wenlong Sun, Patrick Shafto, and Wenlong Sun. “ThinkIR : The University of Louisville ′ s Institutional Repository Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering”. In: (2019).

[48] Zhoubao Sun, Lixin Han, Wenliang Huang, Xueting Wang, Xiaoqin Zeng, Min Wang, and Hong Yan. “Recommender systems based on social networks”. In: *J. Syst. Softw.* 99 (2015), pp. 109–119. ISSN: 01641212. DOI: 10.1016/j.jss.2014.09.019. URL: http://dx.doi.org/10.1016/j.jss.2014.09.019.

[49] Jeffrey Travers and Stanley Milgram. *An Experimental of the Small World Problem*. 1969.

[50] Sotiris Tsioutsiouliklis, Evaggelia Pitoura, Panayiotis Tsaparas, Ilias Kleftakis, and Nikos Mamoulis. *Fairness-Aware Link Analysis*. Vol. 1. 1. Association for Computing Machinery, 2020. ISBN: 9781450383127. DOI: 10.1145/3442381.3450065. arXiv: 2005.14431.

[51] Twitter. “Q1 2021 Letter to shareholders”. In: (2021), pp. 1–20. ISSN: 00178012. URL: https://s22.q4cdn.com/826641620/files/doc_financials/2021/q1/Q1’21-Shareholder-Letter.pdf.

[52] Wenpu Xing and Ali Ghorbani. “Weighted PageRank algorithm”. In: *Proc. - Second Annu. Conf. Commun. Networks Serv. Res.* (2004), pp. 305–314. DOI: 10.1109/dnsr.2004.1344743.

[53] L Zhang and W Tu. “Six Degrees of Separation in Online Society”. In: *Journal.Webscience.Org* (2009), pp. 1–5. URL: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Six+Degrees+of+Separation+in+Online+Society#4.

[54] Zhongzhi Zhang, Tong Shan, and Guanrong Chen. “Random walks on weighted networks”. In: *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.* 87.1 (2013), pp. 1–7. ISSN: 15393755. DOI: 10.1103/PhysRevE.87.012112. arXiv: 1212.5998.

colophon

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

`https://bitbucket.org/amiede/classicthesis/`

Final Version as of August 23, 2021 (`classicthesis` version 4.4).