Université de Montréal

Mitigating Popularity Bias in Graph Based Music Recommendation

par

Rebecca Salganik

Département de mathématiques et de statistique
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

23 novembre 2023

Mitigating Popularity Bias in Graph Based Music Recommendation

présenté par

Rebecca Salganik

Résumé

La dernière décennie a apporté avec elle une vague de technologies innovantes, modifiant la manière dont le contenu créatif est créé, consommé et catégorisé. Et, à mesure que nos interactions avec les contenus multimédias créatifs se déplacent vers les plateformes en ligne, la quantité de contenu sur ces plateformes a nécessité l'intégration d'un guidage algorithmique dans la découverte de ces espaces. De cette façon, les algorithmes de recommandation qui guident les interactions des utilisateurs avec diverses formes d'art ont été jetés dans le rôle de gardiens et ont commencé à jouer un rôle de plus en plus influent dans l'élaboration de la création de contenu artistique.

Le travail présenté dans les chapitres suivants fusionne trois grands domaines de recherche : l'apprentissage de la représentation graphique, la recherche d'informations musicales et l'équité appliquée à la tâche de recommandation musicale. Alors que l'influence des systèmes de recommandation continue de s'étendre et de s'intensifier, il est crucial de prendre en compte les effets en aval que les choix de conception peuvent avoir sur l'écosystème plus large de la création artistique. Ces dernières années, l'intégration des réseaux sociaux dans la tâche de recommandation musicale a donné naissance aux réseaux neuronaux de graphes (GNN), une nouvelle architecture capable de faire des prédictions sur les structures de graphes. Parallèlement aux gains miraculeux que les GNN sont capables de réaliser, bon nombre de ces systèmes peuvent également être la proie de **biais de popularité**, les forçant à privilégier le contenu grand public par rapport à des éléments potentiellement plus pertinents, mais de niche ou nouveaux. S'il n'est pas maîtrisé, ce cycle négatif peut perpétuer les disparités de représentation entre la musique d'artistes, de genres ou de populations minoritaires. Et, ce faisant, les disparités dans la visibilité des éléments peuvent entraîner des problèmes à la fois du point de vue des performances et de la société.

L'objectif de la thèse est l'atténuation du biais de popularité. Premièrement, le travail formalise les liens entre l'équité individuelle et la présence d'un biais de popularité parmi les contenus créatifs. Ensuite, nous étendons un cadre d'équité individuelle, en l'appliquant au domaine de la recommandation musicale. Le cœur de cette thèse s'articule autour de la proposition d'une approche basée sur l'équité individuelle et sensible au domaine qui traite le biais de popularité dans les systèmes de recommandation basés sur les réseaux de neurones graphiques (GNN). L'un des éléments clés de ce travail est notre capacité à ancrer notre notion d'équité dans le domaine musical. Afin de faciliter cette prise de conscience du domaine, nous effectuons une augmentation étendue des ensembles de données, en prenant deux ensembles de données de recommandation musicale à la pointe de la technologie et en les augmentant avec de riches fonctionnalités multimodales au niveau des nœuds. Enfin, nous fondons notre évaluation sur le démarrage à froid, montrant l'importance des méthodologies inductives dans l'espace musical.

Mots clés : Équité, recommandation musicale, apprentissage de la représentation graphique

Abstract

The last decade has brought with it a wave of innovative technology, shifting the channels through which creative content is created, consumed, and categorized. And, as our interactions with creative multimedia content shift towards online platforms, the sheer quantity of content on these platforms has necessitated the integration of algorithmic guidance in the discovery of these spaces. In this way, the recommendation algorithms that guide users' interactions with various art forms have been cast into the role of gatekeepers and begun to play an increasingly influential role in shaping the creation of artistic content.

The work laid out in the following chapters fuses three major areas of research: graph representation learning, music information retrieval, and fairness as applied to the task of music recommendation. In recent years, graph neural networks (GNNs), a powerful new architecture which enables deep learning approaches to be applied to graph or network structures, have proven incredibly influential in the music recommendation domain. In tandem with the striking performance gains that GNNs are able to achieve, many of these systems, have been shown to be strongly influenced by the degree, or number of outgoing edges, of individual nodes. More concretely, recent works have uncovered disparities in the qualities of representations learned by state of the art GNNs between nodes which are strongly and weakly connected. Translating these findings to the sphere of recommender systems, where nodes and edges are used to represent the interactions between users and various items, these disparities in representation that are contingent upon a node's connectivity can be seen as a form of **popularity bias**. And, indeed, within the broader recommendation community, popularity bias has long been considered an open problem, in which recommender systems begin to favor mainstream content over, potentially more relevant, but niche or novel items. If left unchecked these algorithmic nudged towards previously popular content can create, intensify, and enforce negative cycles that perpetuate disparities in representation on both the user and the creator ends of the content consumption pipeline. Particularly in the recommendation of creative (e.g. musical) content, the downstream effects in these disparities of visibility can have genuine economic consequences for artists from under-represented communities. Thus, the problem of popularity bias is something that must be addressed from both a technical and societal perspective. And, as the influence of recommender systems continues to spread, the effects of this phenomenon only become more spurious, as they begin to have critical downstream effects that shape the larger ecosystems in which art is created.

Thus, the broad focus of thesis is the mitigation of popularity bias in music recommendation. In order to tailor our exploration of this issue to the graph domain, we begin by formalizing the relationship between degree fairness and popularity bias. In doing so, we concretely define the notion of popularity, grounding it in the structural principles of an interaction network, and enabling us to design objectives that can mitigate the effects of popularity on representation learning. In our first work, we focus on understanding the effects of sampling on degree fairness in uni-partite graphs. The purpose of this work is to lay the foundation for the graph neural network model which will underlie our music recommender system. We then build off this first work by extending the initial fairness framework to be compatible with bi-partite graphs and applying it to the music domain. The motivation of this work is rooted in the notion of discovery, or the idea that users engage with algorithmic curation in order to find content that is both novel and relevant to their artistic tastes. We present the intrinsic relationship between discovery objectives and the presence of popularity bias, explaining that the presence of popularity bias can blind a system to the musical qualities that underpin the underlying needs of music listening. As we will explain in later sections, one of the key elements of this work is our ability to ground our fairness notion in the musical domain. Thus, we propose a domain-aware, individual fairness-based approach which addresses popularity bias in graph neural network (GNNs) based recommender systems. In order to facilitate this domain awareness, we perform extensive dataset augmentation, taking two state of the art music recommendation datasets and augmenting them with rich multi-modal node-level features. Finally, we ground our evaluation in the cold start setting, showing the importance of inductive methodologies in the music space.

Keywords: Fairness, Music Recommendation, Graph Representation Learning

Contents

List of Tables

List of Figures

Liste des sigles et des abréviations

GNN	Graph Neural Network - *Réseau neuronal graphique*
GCN	Graph Convolutional Network - *Réseau convolutif de graphes*
CF	Collaborative Filtering - *Filtrage collaboratif*
MPD	Million Playlist Dataset - *Million d'ensembles de données de listes de lecture*
LFM	LastFM Dataset - *Base de données de LastFM*

Remerciements

Je tiens à exprimer ma gratitude sans fin à la communauté de personnes qui m'ont soutenu tout au long de ce travail.

Tout d'abord, merci aux membres du comité pour le temps, les efforts et l'attention qu'ils ont consacrés à la lecture de cet ouvrage.

Deuxièmement, un grand merci à Vanuk Harrison, Michael Brandstein et William Hamilton pour avoir vu une chercheuse en moi avant que j'en ai vu une en moi-même. Le soutien que j'ai reçu de chacun d'entre vous a changé le cours de ma vie pour toujours et je ne peux exagérer ma profonde gratitude pour cela.

Troisièmement, un tout aussi grand merci aux personnes que je considérerai à jamais comme mes mentors: Andres Ferraro, Lien Michiels, Jian Kang et Nestor Napoles. Sachez que l'exemple que vous m'avez donné vivra à jamais dans mon esprit. Je passerai le reste de ma carrière à faire de mon mieux pour imiter votre confiance, votre curiosité et votre attention.

Quatrièmement, aux incroyables membres du laboratoire qui ont créé un environnement positif et encourageant : Kiarash, Nicola, Rohan et Afaf. Je n'aurais pas pu réussir sans vous. Merci de ne jamais m'avoir poignardé dans la carotide, même lorsque je l'ai expressément demandé.

Cinquièmement, à mes deux collaborateurs phénoménaux : Golnoosh Farnadi et Fernando Diaz. Si quelqu'un m'avait dit tout ce que j'apprendrais de vous, je ne l'aurais jamais cru. Merci du fond du cœur d'avoir cru et investit en moi, je suis éternellement touchée par votre soutien.

Enfin, aux personnes qui donnent tout son sens à ma vie : mon compagnon, Othman, mon meilleur ami, Julien, et mes parents, Marina et Misha. Merci de m'avoir supporté. Je ne le mérite certainement pas, mais je passerai le restant de mes jours à essayer de vous rendre fière.

Acknowledgements

I would like to express my unending gratitude to the bafflingly large community of people who supported me in the duration of this work.

First, thank you to my committee members for the time, effort, and attention they have put into reading this work.

Second, a massive thank you to Vanuk Harrison, Michael Brandstein and William Hamilton for seeing a researcher in me before I saw one in myself. The support I received from each of you changed the course of my life forever and I can't overstate my profound gratitude for this.

Third, an equally massive thank you to the people I will forever consider my mentors: Andres Ferraro, Lien Michiels, Jian Kang, and Nestor Napoles. Please know that the example that you have set lives rent free in my brain. I will spend the rest of my career trying my best to emulate your confidence, curiosity, and care.

Fourth, to the amazing lab members who created a supportive and encouraging environment: Kiarash, Nicola, Rohan, and Afaf. I could not have graduated without you. Thank you for never stabbing me in the carotid, even when I specifically requested it.

Fifth, to my two phenomenal collaborators: Golnoosh Farnadi and Fernando Diaz. If anyone had told me how much I would learn from you, I would have never believed them. Thank you from the bottom of my heart for believing and investing in me, I am eternally humbled by your support.

Finally, to the people who truly give my life all its meaning: my partner, Othman, my best friend, Julien, and my parents, Marina and Misha. Thank you for putting up with me. I definitely don't deserve it but I will spend the rest of my life trying to earn it.

Chapter 1

Introduction

As our interactions with creative multimedia content shift towards online platforms, the recommendation algorithms that guide our interactions with various art forms play an increasingly influential role in shaping how content is created, communicated, and consumed [**13**]. The seemingly endless amount of content, coupled with societal expectations of personalization, have prompted the integration of algorithmic curation [**150**] into almost every media platform [**129, 77**], and particularly music streaming [**51**]. On one hand, algorithm-driven platforms have created opportunities for users to consume diverse and innovative content from far flung corners of the world. However, simultaneously, analysis has shown that continued interaction with algorithmic curation can create negative feedback loops in the form of diminished content diversity [**29, 96, 111**], perpetuation of gender disparities [**56**], and overblown attention towards maintream content [**12, 4, 29**].

Together, these issues have contributed to the growing awareness around the importance aligning machine learning goals with human ethical values, particularly in the recommendation domain [**5**]. Thus, in recent years, the recommendation community has begun the work of translating normative objectives into mathematical ones in order to minimize the disconnect between the desired and actualized outcomes of an algorithmic system. In particular, this work focuses on the phenomenon of **popularity bias** [**5, 28, 84, 122, 144, 32**] which arises when popular items are given exponential visibility to the detriment of niche or novel music. At their core, the purpose of recommender systems is to facilitate meaningful connections between consumers and producers, via the art they create. However, the presence of popularity bias catalyzes two major points of disconnect in relation to this broad directive. First, as shown extensively in the body of literature related to fairness in recommender systems, the presence of popularity bias can profoundly hinder a system's ability to achieve expose users to novel and relevant content [**27, 159, 106, 132, 30, 17, 3**]. Furthermore, the downstream consequences of this bias is explored in a large body of techno-cultural literature showcasing the ways in which disparities in exposure perpetuates negative feedback loops which have incredibly potent and serious negative effects on the lives of artists and the art they are (dis-)empowered to create [**12, 13, 18**]. At the same time, recent works in the field of human-computer interaction have shown that there is a large swath of users who expect curatorial interactions with recommender systems to introduce them to *new* music that is both novel and relevant to their self-reported aesthetic tastes [**125, 128**]. However, a system which is deeply reliant on popularity to perform recommendation is often unable to serve niche, or previously undiscovered content, due to the nature of its biases against items with low levels of previous user interactions.

Thus, our work takes on the task of mitigating popularity bias in the context of facilitating meaningful music recommendation. In our experimentation we consider the implications of this phenomenon on graph-based music recommendation, presenting a novel mitigation strategy that uses ranking based individual fairness to combat popularity bias. We begin by unpacking the intrinsic relationship between popularity, node degree, individual fairness, and ranking. Our method is deeply informed both by recent political movements within the music domain which are associated with artists' remuneration and signals from the industry that echo the need for concrete guidance from the research community on how to effectively apply fairness frameworks that are intuitively relevant to the music domain [**14**]. Thus, one of the key tenants underlying this work is the notion that fairness frameworks should be tailored to the specific needs of their setting, rather than an abstract or generalized formulation. And, as we will explain in later sections, what differentiates our method from those which have been previously proposed in mitigating popularity bias is our explicit focus on formulating fairness from a domain-aware perspective. By grounding our approach in music features, we are able to design a method that has intuitive connections to the underlying problems caused by popularity bias: namely that if two songs were musically similar, this similarity should be represented in their learned representations *irrespective of their popularity.* And, while the work of this thesis is explicitly tailored to the music domain, we hope that these promising findings can inspire future approaches which are grounded in concrete, domain specific attributes that can be applied to other modalities of creative content.

1.1. Contributions

The contributions of this thesis are structured around the following articles presented in the chapters below. The work detailed in Chapter 3: *Analyzing the Effects of Sampling on Individual Fairness* was presented at the **FaccTRec Workshop on Trustworthy Recommendation at the RecSys conference** in 2022. In addition, the work laid out in Chapter 4: *Fairness Through Domain Awareness: Mitigating Popularity Bias For Music Discovery* has been submitted to a conference and is publicly available (see `https://arxiv.org/abs/2308.14601`).

1.2. Thesis Layout

The work completed in the duration of this thesis fuses several major areas of research: music information retrieval (MIR), graph representation learning, recommendation, and machine learning fairness. Thus, in Chapter 2 we introduce the relevant literature necessary for contextualizing the work done in later chapters. Having introduced the necessary background information for interpreting our work, we present Chapter 3 where we test the compatibility of *REDRESS* with our underlying recommender architecture, showing that, in applying the framework to a sub-sampling based GNN we are able to better improve fairness performance. Then, in Chapter 4 we apply this framework to design a domain-aware graph-based music recommendation model. The goal of our method is to facilitate discovery by designing an algorithm that is able to build rich, multi-modal and expressive representations for all items, irrespective of their popularity.

Chapter 2

Background

The purpose of this chapter is to provide necessary background information for contextualizing the work completed in the duration of this thesis. First, in Section 2.1, we lay out the various methodologies used by the music information retrieval (MIR) community for learning representations of musical content. The information laid out in this section is intended as a basis for understanding our feature selection and general representation learning strategies. In particular, our work is heavily informed by the approaches taken by various competitors in the *Automatic Playlist Continuation Challenge* hosted by Spotify in RecSys 2018 [**31**]. The hybrid nature of the various approaches presented in this challenge solidified the importance of domain-awareness and hybrid recommendation for the field of music recommendation. Then, in Section 2.2 we present a brief introduction to graph representation learning. The purpose of this chapter is to show the trajectory of various concepts and their contributions to the formulation of both our recommendation task and the mitigation strategy. Following this, in Section 2.3, we acquaint the reader with various important elements of recommender system research that contextualize the approach for designing and evaluating our music recommender architecture. Finally, in Section 2.4 we present both the notion of fairness as applied to machine learning and the specific fairness criteria that are later employed in our strategy for mitigating popularity bias.

2.1. Music Representation Learning

The question of how to distill musical content into vector representations lies at the core of all intersections between music and technology, generally termed *music information retrieval* (MIR). In addition to the technical challenges of representation learning, MIR tasks are intrinsically more complicated due to the difficulty of capturing the abstract and multi-faceted nature of artistic content. This is because music is not just a sonic element, it can be seen through a multitude of diverse perspectives - it is a cultural product, an artifact of an individual's artistic expression, a form of communication, an experience, and much more.

Thus, the field of MIR has a long history of interdisciplinarity, fusing signal processing, musicology, cognitive science, and many other disciplines. And, the variety of approaches can be attributed to the fact that architectural design choices are heavily influenced by the downstream outcomes necessitated by a particular application. As such, various attempts to codify music have harnessed a variety of information streams in their methodologies for generating representations. For example, using music notations [**124, 177, 158**], sonic elements [**126, 39, 92**], listening patterns [**134, 64, 136, 95**], metadata [**121, 119, 118**], or a combination of these feature groups.

As we show in later sections our work is heavily influenced by previous research done in this field expanding upon the modalities of information that is being used to represent musical items. Concretely, we collect an aggregate collection of features containing sonic signals, artist demographics, popularity metrics of individual songs, embeddings of album artworks related to songs, embeddings of lyrics and track names, and extract collaborative filtering signals from the playlist-track interactions.

Tying the concepts from generalized MIR to the specific needs of a recommendation setting, the various available modalities for representing musical content are often distinguished into two broad groups: content-based (or descriptive) and context-based (or consumptive) representational methodologies. In the following section we will introduce these approaches and later, in Section 2.3 we will explain how they are harnessed in a music recommender system.

2.1.1. Content-Based Embeddings

The premise of content-based approaches is simple - musical items should be represented by the content that defines them. Most often, content-based approaches define musical items based on their acoustic elements. Thus, content-based approaches are often deeply tied with advances in signal processing. For example, many methods harness either high-level features such as melody, harmony, or rhythm or low-level features that extract frequencies within audio such as mel spectrograms and Mel-Frequency Ceptral Coefficients (MFCC) [**140**]. Currently a majority of the state of the art approaches in this domain harness various deep learning paradigms. For example [**39, 126, 66**] build on recent advancements in computer vision to extract learn item representations from the visualizations of spectrograms. Several methods have also focused on using contrastive learning paradigms to learn latent feature representations [**142, 166**]. Finally, innovations within the nascent field of music generation has found that using VAEs to learn musical representations for generative tasks can also be used in content-based representation learning [**45, 60**].

Ultimately, the strength of this methodology comes from its ability to capture rich information about each individual musical item in a dataset irrepsective of user engagements with it. As such, it doesn't struggle from issues of sparsity that arise, when for example, a new song has been uploaded to a database and has yet to be listened to by a large number of users. However, at the same time, such methods are unable to capture the aggregate listening patterns, and are thus unable to extract the global patterns and cultural interactions with musical items that are formed by societal movements. In this way, their granularity is both a strength and weakness.

2.1.2. Context-Based Embeddings

Another, complementary stream in music representation which is particularly relevant to music recommendation expresses musical items in relation to the consumption patterns in which they are interacted with by listeners. Thus, purely context-based methodology requires interaction data to define musical items using a music agnostic approach. Crucially, unlike content-based embeddings which require some form of metadata associated with individual musical items to perform representation learning, this method harnesses various methods of *collaborative filtering* (see Section 2.3.3.1 for more details and relevant citations) to extract latent item representations by aggregating patterns among listeners who interact with a musical database.

Given the domain-agnostic perspective of this methodology, it is not unusual to see recommender systems taken from other domains and applied to music in order to extract musical representations. For a large collection of innovative collaborative filtering approaches that are applied specifically to music, we direct the reader to [**171**].

One of the drawbacks of these approaches is their reliance on interaction data between an item and song. As such, they are particularly sensitive to the popularity of items and poorly suited to tasks in which new items are being integrated into existing databases. However, due to their ability to capture aggregate listening patterns, they have been shown to outperform content based methods on many recommendation tasks [**145**].

2.1.3. Hybrid Approaches

Given the benefits and drawbacks that are intrinsic to both content and context based approaches, recent work in the field has presented methods for fusing these methodologies together. As explained in later sections, this need is particularly poignant in the recommendation space, where the purpose of generating representations for musical items is to enable comparisons of similarity based on both individual and global listening patterns.

There are many different hybrid approaches for music representation learning. For example [**55**] combine low-level acoustic features with collaborative filtering to perform music recommendation. Alternatively, [**8**] design an ensemble architecture which combines exclusively content-based and context-based methods and then weighs their respective scoring functions to generate playlist recommendations. Alternatively, [**152**] use a two-stage architecture that combines Weighted Regularized Matrix Factorization (WRMF) [**79**] with XGBoost [**35**], a gradient boosting learning to rank model.

2.2. Graph Representation Learning

The study of networks, or graph-like structures, has a long history within various fields both in the technical and social sciences [**10, 115**]. The benefits of these structures arises from their ability to express connections between data points in addition to the individual data points themselves. Thus, they are extremely well suited to tasks where individual data points are influenced by one another. For example, drug discovery [**61**] and social networks [**24**] have been very heavily explored by the graph research community. Within recent years, this architecture has also been applied to the field of recommendation where harnessing social structures that emerge within user groups has yielded state-of-the-art results in many different domains [**154, 58**]. In this chapter we begin with a brief description of the historical trajectory of graph learning. We will introduce several graph neural network architectures and present the most prominent architectures that have been applied to the recommendation domain. Due to the breadth of this field, we will focus our scope on methods that lay the foundation for the work presented in this thesis. We direct readers towards the following works for more detailed presentation of graph learning [**133, 69, 115, 10**].

2.2.1. Notations

We begin by formalizing some notation and providing definitions that form the backbone of this research field.

Definition 2.2.1. Graph: A *graph* is defined as $G = (\mathcal{V}, \mathcal{E})$, consisting of:

(1) a set of *nodes*, $\mathcal{V}$ that can have multiple types. For example, in our setting, we have both user and item nodes such that $\mathcal{V} = \mathcal{U} \bigcup \mathcal{I}$ where $\mathcal{U}$ are users and $\mathcal{I}$ are items.

(2) a set of *edges*, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ that connect two nodes, V_i, V_j and are meant to represent interactions between the nodes. In our setting we work with unweighted and undirected edges.

Note: in our setting we consider $n = |\mathcal{V}|$ to be the total number of nodes and $m = |\mathcal{E}|$ to represent the total number of edges.

Definition 2.2.2. Adjacency matrix: A *adjacency matrix*, $A \in [0{,}1]^{n \times n}$, is defined based on the edges connection in graph, $\mathcal{G}$. Thus, for each entry, $A[i{,}j]$:

$$A_i j = \begin{cases} 1 & \text{if } \exists e_{ij} \in E, \text{ between nodes } n_i, n_j \in V \\ 0 & \text{otherwise} \end{cases}$$

Note: In our setting we remove *self loops* in that the diagonal entries, $A[i{,}i] = 0$.

Definition 2.2.3. Neighbourhood: A *neighbourhood* is a set of *neighbour* nodes, $n_v \in \mathcal{N}(u)$, such that all the nodes, n_v, share an edge, e_{vu}, with a central node, n_u. In relation to an adjacency matrix, A, a *neighborhood* can be defined as:

$$\mathcal{N}(u) = \{v \in \mathcal{V} : A_{vu} = 1\}.$$

Note: we will use the term *k-hop neighbourhood* to refer to neighbourhoods that are formed by looking at the edges that are *k-steps* removed from a central node. For example the definition presented above would be considered a *one-hop neighbourhood*.

Definition 2.2.4. Degree Matrix A *degree matrix*, $D \in [0, n]^{n \times n}$, is defined based on the outgoing edges, $e_{uv} : v \in N(u)$ for each node u. Each entry in D is equivalent to the sum of a row in an adjacency, A.

Definition 2.2.5. Laplacian Matrix The *Laplacian* is an important matrix in the field of graph learning that forms the basis of many different theoretical and empirical innovations within the field. Broadly, the Laplacian summarizes many important properties of the graph [**141**]. In its most general form, the *Laplacian matrix* is defined as:

$$L = D - A$$

There is also a normalized version:

$$L_{\text{norm}} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$$

Definition 2.2.6. Transductivity and Inductivity in Graphs One key question when designing the evaluation schema for a GNN is whether the model is expected to perform in a *transductive* or *inductive* setting. This division refers to the allocation of held-out information that is used in the test set. An *inductive* model is one that is able to perform graph learning tasks on a held out subgraph (that is, a portion of the graph is never used in the training regime). Meanwhile, a *transductive* model requires that all of the nodes in a graph are visible both during training and testing (even if edges between them are masked during various phases of the representation learning methodology). As we will explain in Section 2.3.2.3, this setting has important downstream consequences on the applicability of various GNN architectures to recommendation tasks.

2.2.2. Tasks in Graph Learning

Generally, graph learning is organized with respect to various tasks.

(1) **Edge (Link) Prediction**: In this particular task, the objective is to recover masked edges to predict the existence of connections between the nodes of a graph. This task can be both transductive and inductive (see above), meaning that the masked edges can be connecting nodes that were present in training or nodes that were not present in the training set.

(2) **Node Classification**: In this particular task, the objective is node classification which involves predicting some label y for a node $v \in \mathcal{V}$.

(3) **Graph Clustering**: In this particular task the objective is to perform some form of grouping over the nodes of the graph such that the predicted clusters share some unifying property. Traditionally, this method has been used heavily to discover communities in social and citation networks.

(4) **Graph-Level Prediction**: In this particular method, the objectives of *node classification* are abstracted to the level of a graph.

In general, the broad directive of graph learning is the extraction and analysis of patterns within a graph structure [**10**]. These various methods have a broad range of expressiveness, starting from statistical methods that extract individual descriptors from a graph and extending to complex methods used in deep learning.

2.2.3. Statistical Methods

There are several principles that play an important role in the structure, definition, and categorization of a graph. Two crucial principles that play an important role in the work of thesis are node degree and node centrality.

2.2.3.1. Node Degree. As described in Section 2.2.1, a graph, $\mathcal{G}$, consists of a node set, $v \in \mathcal{V}$, and edge set, $e \in \mathcal{E}$. The degree of a node, u, can be described as the number of outgoing edges, $e_{uv} : v \in N(u)$. More formally:

$$d_u = \sum_{v \in V} A[u, v] \tag{2.2.1}$$

2.2.3.2. Node Centrality. While node degree is able to capture information about an individual node, it does not express the connected-ness of a node with respect to the remainder of the graph. Thus, the purpose of the node centrality measure is to contextualize the degree of a node in relation to the degree of other nodes and their importance in the rest of the graph. In particular, we define a node's centrality via a recurrence relation in which the node's centrality is proportional to the centrality of its neighbours. More formally:

$$c_u = \frac{1}{\lambda} \sum_{v \in V} A[u,v] c_v, \ \forall u \in V, \tag{2.2.2}$$

As we will show in later sections, the degree of a node can play an important role in the quality of the representation learned with respect to this node. This is because many of the recommender architectures introduced in Section 2.3.4 are sensitive to the distribution of a node's neighbourhood when learning its representation.

2.2.4. Kernel Methods

While individual metrics such as those presented in the previous section can provide low-level information about a node, they are often not expressive enough to truly perform well on any of the tasks outlined above. There is a large body research devoted to the study of graph kernels [**15, 151**]. Here, we present two kernel methods that play an important role in laid the graph neural networks (GNNs) that contribute to the work of this thesis.

2.2.4.1. The Weisfeiler-Lehman Kernel. The premise of this kernel is to extract node-level features that are enriched with information from their local neighbourhood nodes [**139**]. By aggregating over neighbourhoods of nodes in an iterative process, the kernel is able to extract information about various communities with the graph and integrate this with the information about individual nodes contributing to these communities. The general procedure of this kernel is:

(1) Assign an initial label $l(0)(v)$ to each node.
(2) Iteratively assign a new label to each node by aggregating over the labels of its neighbour nodes: $l(i)(v) = l(i-1)(u) \forall u \in N(v))$
(3) After running K iterations of gathering and relabeling, we have a label $l(K)(v)$for each node that summarizes the structure of its K-hop neighborhood.

2.2.4.2. Random Walk Kernel. Another deeply influential kernel is the random-walk kernel proposed by Kashima et al. [**90**] which involves running random walks over the graph and then counting the occurrence of different node sequences. A random walk on a graph is a process that begins at some vertex, and at each time step moves to another vertex [**141**]. This kernel method has served as a bridge between kernel and spectral methods because it can be formalized using a variation of the Laplacian matrix:

$$L_{\mathrm{RW}} = D^{-1}L$$

One of the important properties of this approach is its ability to achieve a stable distribution over the node representations despite the randomness of the walks [**141**]. The notion of random walk based embeddings has been given significant attention since the proposal of this kernel method.Most notably, Leskovic et al. [**100**] use the Personalized PageRank algorithm [**120**] to compute a one-hot indicator vector for node u that gives the stationary probability that random walk starting at node u visits node v. Or similarly, a method converted from the natural language processing, Node2Vec [**67**] uses an encoder/decoder architecture to learn the probability of visiting u on a length-T random walk starting at v.

Together these methods contribution to the design and implementation of GraphSAGE and, later, PinSage, two graph neural network architectures that are discussed in detail within the following sections of this thesis. Crucially, the ideas of (1) harnessing a node's neighbourhood into its representational learning paradigm and (2) the use of random walks to summarize the connections between various nodes both play pivotal roles in the relevance of a network structure on consumption patterns.

2.2.5. Spectral Methods

Another pivotal concept which intersects heavily with the notion of kernel methods is that of applying spectral analysis to graphs.

As defined in Section 2.2.1, a graph can be defined using the interplay between the Adjacency and Laplacian matrices. In particular, decomposing the Laplacian matrix into its eigenvalues can give insight into the connections of a graph such that it can be partitioned into various clusters.

2.2.5.1. Graph Clustering. This notion of graph clustering has allowed researchers to develop generalized techniques for learning low dimensional embeddings of a node based on examining the K smallest eigenvectors of the Laplacian. The general algorithm follows the following approach:

(1) Find the K smallest eigenvectors, $u_0, u_1, ...u_k \in \Re^{|V| \times 1}$, of the Laplacian, L.
(2) Define a matrix, $U \in \Re^{|V| \times (k-1)}$ such that the columns of U are the eigenvectors, $e_0, e_1, ...e_k$.
(3) Consider each row of the matrix as the representation, z_n for a node $n \in V$.
(4) Run K-means clustering on $z_n, \forall n \in V$

Since its initial proposal, there have been many works that build on this clustering approach. But, at the core of this methodology lies the foundational importance of the eigenvectors of the Laplacian matrix and its ability to express underlying patterns between the nodes in a graph.

2.2.6. Graph Neural Networks (GNNs)

Generally, the methods expressed in earlier sections can be considered *shallow* because they do not contain parameters that can be *learned*, or tuned through deep learning paradigms of gradient-based optimization. Extending the mathematical ideas that form the underpinnings of the previously defined methodologies, are a series of architectures called graph neural networks which apply the paradigms of deep learning to graphs. In this section we will provide an introduction to the methods that are used in later sections of this thesis, however, we leave [**163, 69, 149, 41, 25**] for more extensive details on the state of the art in this domain.

2.2.6.1. Message Passing Paradigm. Most GNN architectures can be intuitively understood from the perspective of their *neural message passing* paradigm [**63**] in which the nodes within a neighbourhood, $v \in N(u)$, of a central node, u, iterative exchange information in the form of vectors to aggregate a localized representation of the central node, u.

The basic structure of a GNN can be abstracted into two functions, *UPDATE* and *AGGREGATE*. Here, *AGGREGATE* refers to the process of gathering the neighborhood nodes and *UPDATE* refers to the procedure used to update a node's embedding with respect to its neighborhood feature set. Thus, the GNN process of training embeddings, is based on the interactions between these two processes. At each iteration k of training, the *AGGREGATE* function takes the embeddings of u's neighbors, $v \in N(u)$, to generate a message which the *UPDATE* function combines with the previous embedding of node u, $h_u^{(k-1)}$, to generate the updated embedding $h_u^{(k)}$. For more details on the specifics of this formulation, please see Section 3.3.

2.2.6.2. Graph Convolutional Networks. Graph Convolutional Networks are one of the most prominent graph learning architectures within the larger domain of graph learning. The underpinnings of this method come from applying convolutions (previously popularized by the computer vision domain) to graph structures. One critical aspect of the convolution operation, $*$, is that it can be computed by an element-wise product of the Fourier transforms applied to two functions:

$$(f * h)(x) = F^{-1}(F(f(x)) \circ F(h(x))$$

Thus, by defining the Fourier transform on the graph domain, we are able to simplify the process of applying convolutions to a graph. Generally, these methods can be categorized under two major directions: spectral and spatial convolutions. Spectral GCNs define convolutions through spectral filters by drawing on mathematical principles used in graph signal processing and rely heavily on the decomposition of the Laplacian matrix. Meanwhile, Spatial GCNs define convolutions based on the graph topology and are motivated by principles of information propagation on graphs.

Spectral Methods Spectral methods rely heavily on the decomposition of the Laplacian and adjacency matrices into their eigenvectors. In particular, the Normalized graph Laplacian is a real symmetric positive semi-definite matrix. This means that it can be decomposed into the form $L = U\Lambda U^T$ such that $U = [u_0, u_1, ..., u_{|V|-1}] \in \Re^{|V|\times|V|}$ is a matrix composed of eigenvectors ordered by the magnitude of their eigenvalues and Λ is a diagonal matrix of eigenvalues (*spectra*) where $\Lambda_{ii} = \lambda_i$.

The Fourier transform is generalized to graphs through this eigendecomposition of the Laplacian where the eignevectors U constitute the graph Fourier modes (the complex exponentials of the Fourier series composing the function). Thus, the Fourier transform projects a graph signal x onto an orthonormal basis defined by U:

$$F(x) = \hat{x} = U^T x \tag{2.2.3}$$

and the inverse Fourier is defined by:

$$F^{-1}(\hat{x}) = U\hat{x} \tag{2.2.4}$$

.

Going back to the relationship between convolutions and the Fourier transform presented above, we can now apply convolutions to the graph domain using the graph Fourier transform defined a few lines above. Thus, a graph convolution, $*_G$, of the an input signal, x, with a filter, $g \in \Re^n$ can be defined as:

$$x *_G g = F^{-1}(F(x) \odot F(g)) = U(U^T x \odot U^T g) \tag{2.2.5}$$

where $\odot$ denotes an elementwise product. We can further simplify this equation if we consider the filter g in the Fourier domain as $U^T g \in \Re^n$ and diagonalize it as $g_\theta = diag(U^T g)$:

$$x *_G g = U(U^T x \odot g_\theta) = U g_{T_x} \tag{2.2.6}$$

This forms the abstract blueprint for various spectral GCN implementations, which will differ on the basis of their filter selection. The Graph Convolutional Network (GCN) method proposed by Kipf and Welling [**93**] is the most canonical of these methods due to its improvements in computational efficiency. This method defines the filter and the resulting convolution layer as:

$$x *_G g_\theta = \Theta(\bar{A})x \tag{2.2.7}$$

where $\bar{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ with $\tilde{A} = A + I$, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, and Θ is a set of learnable parameters.

Since its proposal, this method has become incredibly influential and spawned a variety of extensions both in graph representation learning and beyond (see Section 2.3 for further details). Within the graph domain, a notable extension is Adaptive Graph Convolutional Network (AGCN) [**101**] which is able to adaptively select the neighbourhood to learn hidden structural relations unspecified by the graph adjacency matrix. Alternatively, Dual Graph Convolutional Network (DGCN) [**181**] introduces a dual graph convolutional architecture which learns both spatial connections and interdependencies in the feature space among connected nodes.

Spatial Methods Spatial convolutions can be seen as a relatively direct translation from the convolutions in CNNs to the graph domain. CNN's apply $n \times n$ filters over localized sections of an image. Drawing on this principle, these filters in the graph space can are analogous to the neighbourhood surrounding a node. Drawing on the message passing paradigm, a spatial convolution can be understood through the *AGGREGATE* and *UPDATE* defined previously where aggregation is the construction of a spatial filter and update is the application of it to a particular node of the graph. There are many variations of spatial GCNs, differing on the basis of their kernel construction and update methodology. In later sections we work extensively with the GraphSAGE [**71**] model. The message-passing paradigm of GraphSAGE differ slightly in that it relies on randomly sampling p neighbours from the surrounding neighborhood (i.e., $N_p(u)$) rather than interacting with a fixed set of neighbours. For a more formal definition of the learning paradigm, please see Section 3.3. As we will address in later sections, since its introduction, GraphSAGE has been extended to become the canonical recommender system, PinSage [**170**], which will become the backbone recommender model used in our music recommendation task.

2.3. Recommender Systems

In this section we present the task of recommendation, beginning with the generalized setting and then building on the previous sections to explain how it differs in the music domain. We begin by presenting the relevant notations that are used to formalize the recommendation setting. Then, we present a variety of settings in which recommender systems can be evaluated and the metrics which are relevant for these evaluations.

2.3.1. Notations

Definition 2.3.1. Interaction Matrix An *interaction matrix* is a matrix, $\mathcal{M}$ which is used to codify the interactions between a set of users, U, and a catalogue of items, I. There are several versions of this matrix:

(1) Binary Interactions: often, the interactions between users and items are codified in a binary manner. In this case, the entries of the interaction matrix, $\mathcal{M} \in [0,1]^{|U| \times |I|}$, are represented with binary values. As we explain in later sections (See Implicit and Explicit Feedback, Section 2.3.2.1) these values can come from implicitly inferred or explicitly stated user feedback.

(2) Ratings: in the case that user interactions have the properties of ordered relations, that is: $r_1 < r_2 < ...r_n$, rather than using binary entries, the matrix will contain numerical values. In this case, the entries of an interaction matrix, $\mathcal{M} \in [r_1, r_n]^{|U| \times |I|}$, are represented by the ratings given by a user to an item.

2.3.2. Recommendation Settings

The fundamental goal of recommendation tasks is to recommend relevant items for a user based on their previous preferences. However, the flexibility in defining user/item interactions, harnessing external feature sets, and generating recommendations have paved the way for many different recommender architectures.

2.3.2.1. Implicit and Explicit Feedback. In order to facilitate the generation of relevant recommendations, recommender systems must extrapolate the patterns present in previous user interactions with an item set. One of the crucial design choices that is implicitly made in dataset selection is the format in which feedback was gathered. The most common form of user review is called *implicit feedback*. This method is used when item ratings are based on whether or not a user has interacted with them. They are termed implicit because a negative interaction and the lack of an interaction will have the same rating (the lowest possible one being 0). Thus, implicit feedback can lead to noisy or presumptuous learning that isn't necessarily representative of a user's true feelings about an item. The other, more concrete form of feedback, is termed *explicit feedback*. Unlike implicit feedback, in which interaction data is passively inferred from a user's browsing patterns, explicit feedback requires direct responses from a user. For example, consider a user who is interacting with the radio function on a streaming platform. As the recommender system provides a continuous stream of music to listen to, a user occasionally responds to one of the songs with a skip. However, the lack of a skip does not necessarily imply a positive interaction. Alternatively, consider a user who is constructing their own playlist. In actively selecting songs for this playlist, they are explicitly marking these songs as those with which they have had positive interactions.

2.3.2.2. External Feature Sets. As mentioned in earlier sections relating to representation learning in music (see Section 2.1), many recommender systems can also be categorized along the lines of whether they use content-based or context-based information when generating item and user level representations. Thus, a crucial question that directly impacts the design of a recommender system is whether it uses interaction data (context-based), individual feature sets (content-based) or a fusion of the two (hybrid) when learning representations. As we will present in later sections, this decision have important consequences on the settings to which a particular architecture may be suited.

2.3.2.3. Cold Start. The term *cold start* is an umbrella term that refers to the evaluation setting in which a recommender system is evaluated on its ability to generate relevant recommendation based on users (or items) that were not present in the training set (see Figure 2.1 for visualization). The purpose of evaluating a recommender system in the cold start setting is to simulate a dynamic environment in which the catalogue of items or collection of users changes over time. Evaluating a recommender system in this setting can better approximate its future performance in an online setting. However, due to the inductive nature of this setting, there are many models which are simply unable to generalize to this setting. And, particularly, in the music setting where new artists and songs are added to platforms on a daily basis [**81**], this problem has received particular attention [**137**].

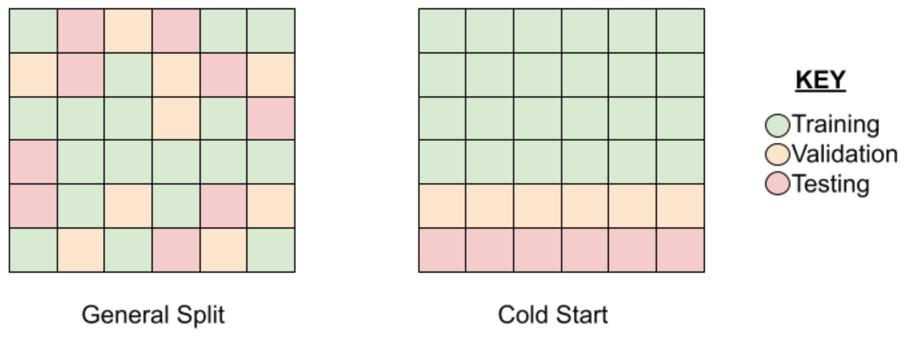

Figure 2.1 – Cold Start Evaluation. Consider an interaction matrix of size *user* × *item*. In the classic recommendation setting, the training/validation/testing splits are performed by randomly sampling from matrix. Meanwhile, in the cold start setting, the splits are preformed by randomly sampling over users. This is intended to simulate an online setting in which new users are being integrated in a platform.

2.3.3. Foundational Methods for Recommendation

We will now present some important foundational models that form the basis of future work in the recommendation space.

2.3.3.1. Collaborative Filtering. In the most general case, recommendations are made on the basis of an interaction matrix where the task of performing recommendation is synonymous with matrix completion. The various methodologies presented to engaging with this task are often referred to as collaborative filtering (CF). Generally speaking, methods that perform CF have two key components: (1) embedding generation, in which they learn vector representations of users and items, and (2) interaction modeling, in which interaction patterns are reproduced using the learned embeddings. One of the most popular approaches in this category is called matrix factorization. This method relies on the rank decomposition theorem to split an interaction matrix into two smaller matrices that contain *latent factors* representing both the users and the items. Multiplying these latent factors recovers entries that were not present in the initial interaction matrix, thus forming the basis of future recommendations.

2.3.3.2. Bayesian Personalized Ranking. Another prominent context-based method that has laid the foundation for future innovations in the recommendation space is Bayesian Personalized Ranking (BPR) [**130**]. Unlike matrix factorization, which aims to make predictions based on individual interactions between a user and an item, BPR harnesses the power of contrast between two items. One of the major contributions of BPR is the notion of performing training in triplets such that each training instance can be expressed as:

$$x = \{(u,i,j) : i \in I_u^+, j \in I \setminus I_u^+\}$$

where u represents a user, i, a positive example of an item liked by user u and j, a negative example of an item disliked by user u. This notion shows itself in many forms throughout the machine learning community, under many different variations, such as triplet loss [**95**], max-margin loss [**170**], Siamese loss [**126**], and many more.

2.3.3.3. Learn to Rank. Another important architecture that plays a foundational role in the work presented in later sections of this thesis is the *learn to rank* paradigm. This category of architectures was originally designed by the retrieval community in order to calculate the order in which results should be presented with respect to an individual search query. The basis of this research is based heavily on findings from the human-computer interaction community indicating that the order in which items are presented has a strong effect on their likelihood of being interacted with [**46**]. Thus, in this setting, given a search query, q, and a set of documents, $d \in D$, a ranking model, $\mathcal{M}$, is trained using query, document pairs, $x = (q,d)$, such that it can predict their relevance score to users, $s = f(x)$. There are many different approaches that integrate various deep learning methods into their architecture design. For a detailed survey, we suggest [**73**]. In this work, we focus on the architecture of LambdaRank [**20**], presented in 2010 and still extremely relevant today. The general methodology of the LambdaRank model involves learning ordering between pairs rather than individual relevance score. More concretely, given a query q and two potential documents, $d_k, d_j \in D$, where the relevance of a document, d_i, to a query, q, is expressed as $s_i = f(x_i), x_i = (q, d_i)$, the learn to rank model will learn the probability that $P_{kj} = P(s_k > s_j)$. This probability can be compared with the true probability distribution, $\hat{P}_{kj}$ (often based on some browsing history), and can then be trained with gradient descent using cross entropy loss:

$$C = -\hat{P}_{kj} log P_{kj} - (1 - \hat{P}_{kj}) log(1 - P_{kj})$$

The crucial element of this methodology is that, in expressing these orderings as probabilities, the model can be trained using differentiation on ranking, which is generally not a differentiable function. We will return to this problem formulation and loss function when discussing the bias mitigation method used in the works of this thesis.

2.3.4. Graph Based Recommendation

In recent years, GNN-based recommender systems have achieved state-of-the-art results on a multitude of recommendation scenarios. One of the key elements to integrating a graph-like structure with the tabular format of interaction matrices, is the creation of a bipartite user/item graph. More specifically, an interaction matrix $\mathcal{M} \in [0,1]^{user \times items}$ can be transformed into a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with the sets, $\mathcal{U}$,$\mathcal{I} \in V$ as the set of user and item nodes, respectively. The entries of the interaction matrix, $\mathcal{M}$, are represented as edges between user/item node pairs, $u_i \in U, i_j \in I$. In this way, matrix completion can be cast a problem of link prediction (see Section 2.2 for details) which consists of inferring the existence of an edge between a user/item node pair, $u_i \in U, i_j \in I$.

In this section we summarize some of the canonical models that form the basis for future innovations in the field. Since the goal of this section is to lay the context for our proposed work, we select the models which are most pertinent. We draw on a series of surveys for synthesizing these architectures. For more details, please see [**58, 165**].

2.3.4.1. Graph Convolutional Matrix Completion: GC-MC (2018). A canonical graph-based model that contributed to the prominence of GNNs in recommendation is called Graph Convolutional Matrix Completion (GC-MC) [**148**]. In a departure from the state of the art methods that came before it, GC-MC formulated matrix completion as a link prediction task on a bipartite graph. This laid the foundation for the swift integration of GNNs into recommender systems because interaction data that was common to the collaborative filtering paradigm could be represented by a bipartite graph between user and item nodes, with observed ratings/purchases represented by links. Thus, innovations in the field of GNNs were operationalized by the recommendation community. The basis of the GC-MC model relies on using a graph autoencoder to generate node-level representations for users and items.

2.3.4.2. PinSage (2018). Another canonical model within the graph recommender community is PinSage [**170**]. Presented as an extension of the GraphSAGE model [**71**] (see Section 2.2.6.2 in for further details). Unlike GC-MC [**148**] which only used 1-hop neighbours to learn node-level representations, PinSage was able to sample k-hop neighborhoods by sampling p neighbours using weighted random walks that most commonly appear in random walks surrounding each training batch node, u.

2.3.4.3. Neural Graph Collaborative Filtering: NGCF (2019). The Neural Graph Collaborative Filtering (NGCF) model [**155**] was presented as an extension of Graph Convolutional Networks (see Section 2.2.6.2 for details) to the recommendation domain. Unlike GC-MC [**148**] which only had one neighbourhood layer, NGCF was able to extract higher connectivity from the graph by stacking multiple convolution layers. In addition, unlike PinSage [**170**] which was intended solely for the purpose of learning item-level representations, NGCF learned both user and item representations at once.

2.3.4.4. LightGCN: LGCN (2020). Proposed by He et al. [**75**] as an extension to the NGCF model [**155**], the Light Graph Convolutional Network (LGCN) has, in recent years, become an important state of the art benchmark. This architecture follows the same structure as the NGCF model but in a lightweight, more compact form, that removes feature transformation and nonlinear activation between layers. Their work shows both empirically and theoretically that these layers do not improve performance and their removal significantly lowers the number of learned parameters.

2.3.4.5. Recent Architectures. In the last few years, as GNNS have become more prominent in the recommendation domain, there have been many innovative works that extend the previously presented models into more specialized problem settings. While the examples provided below are by no means exhaustive, we see them as natural extensions which can be relevant to the work done in later chapters. For example, Dual Channel Hypergraph Collaborative Filtering (DHCF) [**86**], presented in 2020, harnesses the power of hyper-graph GCNs to design a method that is able to distinguish between users and items when learning embeddings. In this way, DHCF is well aligned with the task of music recommendation, where there may be different metadata available for users, playlists, songs, artists, etc. Similarly, the Graph Convolution Machine (GCM) model [**162**], presented in 2022, augments the NGCF architecture with an encoder/decoder framework to accommodate for the complexity of adding contextual data to interactions. Such a method is particularly poignant in the musical space, where the context in which a user is engaging with musical content can have a very strong impact on relevance of various items. Meanwhile, Diversified GNN-based Recommender System (DGRec) [**169**], proposed in 2022, extends the GCN architecture with a novel neighbor sampling technique to improve the diversity of recommended item categories in various e-commerce settings.

2.3.5. Graph Based Music Recommendation

We now present three important works that have used GNNs for music recommendation tasks. Each of these works is selected specifically because, unlike general methods which show their performance on common musical datasets (like LastFM), these methods are designed explicitly for musical settings. It is important to note that each of these important works was published in tandem with one of the major music streaming services. This collaborative pattern is important to note because it showcases one of the key difficulties of engaging with music recommendation tasks from an academic perspective. The proprietary nature of catalogues, feature extraction methods, and methodologies, used by the industry add to the complexity of researching music recommendation.

2.3.5.1. Pandora - GraphSAGE for Artist Similarity (2021). In their work, Korzeniowkski et al. apply the GraphSAGE model (see Section 2.2.6.2 for more details) to the task of learning artist similarity [**95**]. In this setting the graph is constructed using a slice of the proprietary artist catalogue at Pandora as nodes and musicologist annotated connections as edges. In addition to presenting their work, the authors also publicize OLGA, the dataset used in their experimentation. Using the terminology presented in Section 2.1, theirs is a hybrid approach to music representation learning because in addition to using interaction data to generate edges between artists, they harness proprietary music feature extraction methods to provide node-level features. This is important because the integration of music features adds constraints on the graph-based models which can be applied to the setting. In particular, the authors justify their selection of the GraphSAGE model by citing its ability to integrate node-level feature sets. Furthermore, Korzenioswki et al. ground their evaluation setting in the cold start setting (see Section 2.3.2.3 for more details). As they explain in their work, this too participates in their motivation for selecting the GraphSAGE model, citing the inductive nature of this architecture as a crucial element in their methodology. As we will see throughout the works highlighted in this section, the cold start setting is incredibly important for music recommendation tasks and plays a large role in the selection and design of architectures applied to this domain.

2.3.5.2. Deezer - Graph Auto-Encoders for Cold Start Artist Similarity (2021). In another related work, Salha-Galvan et al. design an alternative method for the task of generating artist similarity. Unlike the previous work, the methodology presented in this work is even more explicitly focused on the cold start setting. Furthermore, unlike Korzeniowski et al. [**95**], which works with undirected graphs, Salha-Galvan et al. focus on the directed graph setting. In order to accomodate the unsymmetrical relationships between artists encoded in their training graph, Salha-Galvan et al. use gravity inspired decoders to generate node-level representations. Similar to Korzeniowski et al., this method is also a hybrid method that integrates proprietary features with network connections.

2.3.5.3. Spotify - Multi-task Sampling and Inductive learning on Graphs (2021). In their work, Saravanou et al. approach the broader task of music recommendation through the lens of link prediction on a bipartite playlist-song graph. Thus, unlike the two previously presented works which train on graphs with a single artist node type, Saravanou et al. utilize a bipartite graph to represent the connections between songs and playlists. They posit that, training representations for multiple, interconnected tasks improves the robustness of the learned embeddings. In order to do so, they present 3 tasks:

(1) Playlist Prediction: a binary classification task where the model is evaluated on its ability to predict whether two songs (or tracks) belong to the same playlist.
(2) Genre Prediction: binary classification task where the model is evaluated on its ability to predict whether two songs (or tracks) belong to the same genre.
(3) Acoustic or audio similarity Prediction: a regression task where the model is evaluated on its ability to capture similarities in the music content space.

These three tasks are then unified into a single loss function which is able to apply weighted importance to each individual loss. Similar to Korzeniowski et al. [**95**], Saravanou et al. use the SAGE training paradigm, applying the PinSAGE model [**170**] to its bipartite graph structure.

2.3.6. Discovery: A Subset of the Recommendation Task

As the recommendation research community has grown in size, the overarching objective of designing generalized recommendation techniques has received significant attention. However, in recent years, an interdisciplinary body of work has begun to explore the ”sub-genres“ of this overarching task by introducing peculiarities of specific recommendation objectives within a unique domains and differentiating them from the general setting [**22, 53, 116**]. In parallel, another body of research has begun to codify various consumption patterns that are prevalent among different user bases [**113, 128**]. This research seeks to understand users needs and values when interacting with a particular algorithmic recommendation system within some specific domain and how these needs differentiate the objectives of this system from that of generalized recommendation. Together these research streams have hinted at a future when generalized recommendation methodologies require specialization or tailoring with respect to a particular e-commerce domain.

Within the music domain, recent work has begun to highlight a particularly poignant sub-genre encompassed within the general music recommendation objective: **discovery** [**84, 51, 99, 42, 128, 114, 59**]. Several works have attempted to concretely define discovery such that it can be evaluated and operationalized. Most notably [**59**] present an analysis of user expectations in discovery-oriented recommendation and present a series of evaluation techniques for understanding whether meaningful discovery had been achieved. Meanwhile, [**76**] suggests that the current metrics used for evaluating recommendation are limited on the basis of their ability to understand how a system facilitates novelty. In the same vein, several works have focused on evaluating novelty, using this concept as a proxy for discovery, [**106, 178, 180**]. Even more significant than the work formalizing discovery, has been the work solidifying the connections between discovery and the needs of specific user groups. In their work, Mok et al. [**113**] have made a clear distinction between *manual* and *algorithmic* music consumption. This distinction hinges on the way that new music is discovered, or integrated into a generalized listening pattern. More explicitly, *manual* listening occurs when users manually select songs via the search functionality on a platform. Meanwhile, *algorithmic* listening occurs when songs are queued via some algorithmic system. Implied in the outcomes of this work is the intrinsic connection between discovery and consumption patterns. Raff et al. [**128**] further break down these two categories into four quadrants which are defined on the basis of active to passive interactive patterns between a user and recommender system. In this work, they concretely associate the notion of discovery with distinctive user groups. In addition, Raff et al. [**128**] argue that algorithmic curation is one of the key attributes that affects user satisfaction with a particular platform. Within the sphere of computational musicology and anthropology, similar findings have been validated with qualitative research methods [**51, 125, 42, 114, 59**], with [**114, 117**] cementing the importance of facilitating discovery in music recommendations as a concretely gratifying behavior for users, and [**125**] formalizing algorithmic curation and its role on music discovery as the second most important component contributing to user loyalty.

Together, these bodies of work present an interesting insight into the different needs of users interacting with music streaming platforms. In later portions of this work, we will return to the concept of discovery, connecting it with the prevalence of various algorithmic biases and integrating it into our mitigation technique. Crucially, we wish to note that the notion of discovery is not at odds with the needs of mainstream listening. The purpose of facilitating discovery is not to nudge consumption patterns of mainstream users towards niche content. But rather to shine a light on the distinctive needs of an important tranche of users and formalize it such that it can be integrated into an algorithmic system's objective.

2.4. Recommendation Fairness

2.4.1. General Fairness Methods

The field of fair ML has risen to prominence in the last decade as the proliferation of deep learning has spread throughout the industry and into a large variety of applications and human-facing domains. The growth of this research direction has come from an interdisciplinary mix of perspectives. Most prominently, work in this field has been heavily informed by practices in humanities-adjacent domains such as law, ethics, and sociology combined with more technical fields such as mathematics and engineering. In a general sense, the fairness in machine learning refers to the analysis and mitigation of disparities between the treatment of users with respect to an algorithmic system. These disparities can be measured in many different ways, yielding a variety of different fairness definitions that are often used to assess the outcomes of a particular system. Another important element to the study of fairness in machine learning is the granularity with which users are considered. Currently, there are three major branches of fairness research, each of which takes a unique stance on how fairness should be codified and considered. We will now briefly outline these and then encourage readers to engage with [**11**] for a more thorough knowledge base.

Perhaps the most famous of these is *group fairness* in which users are broken down into groups based on a particular characteristic or attribute and disparities are assessed on a group level [**11**]. The fairness formulations that are most commonly applied in this research area can be traced to principles of discrimination and the legal frameworks associated with it. For example, groups can be defined based on sensitive attributes such as race, gender, or sexuality. Thus, in order to engage with group fairness metrics, practitioners must have access to the sensitive attributes for each data point that is interacted with by an algorithmic system. An alternative approach to fairness is the notion of *counterfactual fairness* [**11**]. Unlike group fairness, which often takes the attributes of users at face value, counterfactual fairness considers the interplay between various individual characteristics. For example, group fairness metrics might assess the likelihood of an applicant being accepted for a loan based on their race. Mitigation attempts harnessing group fairness might attempt to equalize the opportunity for financially stable applicants such that their probability of receiving a loan is irrespective of their racial group. Meanwhile, counterfactual methods would consider how an applicant's racial group might contribute to their financial stability (or lack thereof) and accommodate for differences within groups. Finally, the last common category of fairness definitions is *individual fairness*. This granular approach considers disparities between similar individuals, attempting to mitigate inequalities in their treatment.

2.4.2. Individual Fairness in Graphs

The general premise underlying various definitions of individual fairness is that *similar individuals should be treated similarly (by an algorithmic system)* [**52**]. However, there are many different approaches to defining both the notion of *similar individuals* and *similar treatment* when applied to the machine learning setting. Thus, in order to engage with the principles of individual fairness, machine learning practitioners will often define some function that is able to encode similarity between individuals.

Definition 2.4.1. Individual Fairness Given a model, $\mathcal{M}$ that maps data points, $x \in X$, to outcomes, $y \in Y$, such that $M(x) \to y$, and two distance functions $d(x_i, x_j), D(y_i, y_j)$, the property of *individual fairness* holds when $D\left(M(x_i), M(x_j)\right) \leq d(y_i, y_j)$ [**52**]

Particularly in the space of graph learning, this formulation has had many different interpretations. For example [**88**] defined the InFoRM framework in which individual fairness is upheld when

$$2Tr(Y'L_SY) \leq m\epsilon = \delta \tag{2.4.1}$$

where L_S is the laplacian of the node-node similarity matrix, S, and $Tr(Y'L_SY)$ measures the difference of the mining results, Y and the debiased results, Y' between all pairs of nodes. Meanwhile, [**48**] define the REDRESS framework which considers from a ranking based approach such that for each central node, i, and two graph nodes, j is minimized:

$$L_{j,m}(i) = -P_{j,m} \log \hat{P}_{j,m} - (1 - P_{j,m}) \log(1 - \hat{P}_{j,m}) \tag{2.4.2}$$

where $P_{j,m}$ is formed by calculating the probability that node j is more similar to node i than node m based on the cosine similarity between their input feature vectors and $\hat{P}_{j,m}$ reflects the same probability but applied to the learned node representation embeddings. Finally [**156**] approach fairness from a stochastic perspective to formulate their application of individual fairness to the knowledge graph-based recommendation.

2.4.3. Popularity Bias

2.4.3.1. Defining Popularity Bias. Most broadly, popularity bias refers to a disparity between the treatment of popular and unpopular items at the hands of a recommender system. As such, this term is loosely tied to a collection of complementary terms including exposure bias [**46**], item-based fairness [**156**], user-based fairness [**54**], superstar economics [**12**], long tail recommendation [**108**], the Matthew effect [**112**], and aggregate diversity [**6, 28**]. For a more detailed discussion of the various methods for definition, please see Section 4.2.1.

2.4.3.2. Mitigating Popularity Bias. There has been a lot of work done analyzing and codifying the nature of popularity bias in recommender systems [**23, 82, 84, 32**]. And, there are several axes on which these methods can be classified.

Primarily, the various approaches to mitigating popularity bias can be separated by the general methodology used for their mitigation technique. In particular, these can be **pre-processing**, where modifications are made to debias input data [**17, 88**], **in-processing**, where modifications to the loss function are made to debias the parameters [**132, 159, 179**], or **post-processing**, where the output of a recommender model is re-ordered to mitigate position bias [**3, 108**].

Alternatively, mitigation strategies can also be characterized by their underlying fairness notions. For example, using **group fairness** [**11**] preprocessing techniques sample from the minority population to balance their representation in the training interactions [**17**]. Or, various in-processing methods modify loss functions to regularize for group imbalances [**132, 174**]. Finally, various post-processing methods use re-ranking [**3, 172**]. Alternatively, many approaches use **counterfactual fairness** to disentangle the effect of popularity on a recommender system's score [**159, 179, 176**]. In comparison with group and counterfactual fairness, there has been relatively little work done in relation to **individual fairness**. This is due to the difficulty of defining a similarity through which to define pairs of individuals. In their work, [**30**] define similarity on the basis of relevance, proposing a method in which items are given exposure that is proportional to their predicted relevance. In a more recent work, [**156**] define similarity between learned representations, however, their work is grounded in knowledge graph based recommendation and thus cannot be applied to many recommendation settings.

Chapter 3

Analyzing the Effects of Sampling on Individual Fairness

by

Rebecca Salganik[1], Fernando Diaz[2], and Golnoosh Farnadi[3]

([1]) Mila, Université de Montréal

([2]) Carnegie Mellon University

([3]) Mila, Université de Montréal, McGill University

This was work was accepted at the 5th FAccTRec Workshop on Responsible Recommendation, RecSys 2022.

The main contributions of Rebecca Salganik for this articles are presented.

— Proposal and design of research objectives and experimental methodology (under supervision from collaborators);

— Implementation and development of open source code base for reproducing experimental results;

— Writing the paper as first author;

— Presenting findings at conference.

Co-author, **Fernando Diaz**, helped with insightful feedback throughout the project and paper writing.

Co-author, **Golnoosh Farnadi**, guided the project as principal investigator on the project (PI), providing feedback throughout the project and paper writing.

RÉSUMÉ. Ces dernières années, les méthodes basées sur les réseaux de neurones graphiques (GNN) ont saturé le domaine des systèmes de recommandation. Les gains de ces systèmes ont été significatifs, mettant en évidence les avantages de l'interprétation des données via une structure de réseau. Cependant, malgré les avantages notables de l'utilisation de structures de graphes dans les tâches de recommandation, cette forme de représentation a également engendré de nouveaux défis qui exacerbent la complexité de l'atténuation des biais algorithmiques. Lorsque les GNN sont intégrés dans des tâches en aval, telles que la recommandation, l'atténuation des biais peut devenir encore plus difficile. En outre, la difficulté d'appliquer les méthodes existantes de promotion de l'équité à de grands ensembles de données du monde réel impose des contraintes encore plus sérieuses sur les tentatives d'atténuation. En tant que tel, notre travail vise à combler cette lacune en prenant une méthode existante pour promouvoir l'équité individuelle sur les graphiques et en l'étendant pour prendre en charge la formation par mini-lots ou sous-échantillons d'un GNN, jetant ainsi les bases de l'application de cette méthode à une tâche de recommandation en aval. Nous évaluons deux méthodes GNN populaires: Graph Convolutional Network (GCN), qui s'entraîne sur l'ensemble du graphe, et GraphSAGE, qui utilise des marches aléatoires probabilistes pour créer des sous-graphes pour la formation par mini-lots, et évaluons les effets du sous-échantillonnage sur l'équité individuelle. Nous implémentons une notion d'équité individuelle appelée *REDRESS*, proposée par Dong et al., qui utilise l'optimisation des rangs pour apprendre les nœuds équitables individuels, ou les éléments. Nous montrons empiriquement sur deux ensembles de données du monde réel que GraphSAGE est capable d'atteindre non seulement une précision comparable, mais également une équité améliorée par rapport au modèle GCN. Ces résultats ont des ramifications conséquentes dans les domaines de la promotion de l'équité individuelle, des GNN et, sous une forme en aval, des systèmes de recommandation, montrant que la formation en mini-batch facilite la promotion de l'équité individuelle en permettant aux nuances locales de guider le processus de promotion de l'équité dans l'apprentissage de la représentation.
Mots clés : Apprentissage des graphes, apprentissage des représentations, équité individuelle, échantillonnage

Abstract. In recent years, graph neural network (GNN) based methods have saturated the field of recommender systems. The gains of these systems have been significant, showcasing the advantages of interpreting data through a network structure. However, despite the noticeable benefits of using graph structures in recommendation tasks, this representational form has also bred new challenges which exacerbate the complexity of mitigating algorithmic bias. When GNNs are integrated into downstream tasks, such as recommendation, bias mitigation can become even more difficult. Furthermore, the intractability of applying existing methods of fairness promotion to large, real world datasets places even more serious constraints on mitigation attempts. As such, our work sets out to fill in this gap by taking an existing method for promoting individual fairness on graphs and extending it to support mini-batch, or sub-sample based, training of a GNN, thus laying the groundwork for applying this method to a downstream recommendation task. We evaluate two popular GNN methods: Graph Convolutional Network (GCN), which trains on the entire graph, and GraphSAGE, which uses probabilistic random walks to create subgraphs for mini-batch training, and assess the effects of sub-sampling on individual fairness. We implement an individual fairness notion called *REDRESS*, proposed by Dong et al., which uses rank optimization to learn individual fair node, or item, embeddings. We empirically show on two real world datasets that GraphSAGE is able to achieve, not just, comparable accuracy, but also, improved fairness as compared with the GCN model. These finding have consequential ramifications to the fields of individual fairness promotion, GNNs, and in downstream form, recommender systems, showing that mini-batch training facilitate individual fairness promotion by allowing for local nuance to guide the process of fairness promotion in representation learning.
Keywords: Graph Learning, Representation Learning, Individual Fairness, Sampling

3.1. Introduction

As our consumption habits shift towards online spaces, recommender systems are slowly becoming the gatekeepers between producers and consumers within a wide array of domains. In recent years, the field of recommender systems has become saturated by one particular form of representation learning: graph neural networks (GNNs) [**164, 175, 127, 68, 43, 37**]. The key advantage which distinguishes GNNs from other representation learning methods is their ability to leverage not just the information embedded in the features associated with each data point, but also the information which can be extracted from their interactions [**167**]. However, in parallel to the proliferation of this important architecture, has come a rising level of concern that GNNs come with their own set of unique challenges when it comes to mitigating algorithmic bias. Findings indicating perpetuation or exponentiation of societal biases at the hands of other deep learning models have prompted similar questions in the field of GNNs, specifically as applied to recommender systems [**33**]. As such, the integration of these architectures into recommender systems can have very real, negative consequences on the experiences of both producers and consumers who interact with the recommender system. Without developing bias mitigation strategies for GNN's we cannot properly ameliorate the biases in downstream tasks, such as recommendation.

Within the algorithmic fairness community, there have been several major thrusts, each associated with a broad definition of what it means to engage with fairness from a mathematical perspective [**52, 97, 94**]. The focus of this work is centered on the notion of individual fairness [**52**]. This notion dictates that "similar individuals should be treated similarly by an algorithmic system". Intuitively, promotion of this notion can begin to address problematic discrepancies between the treatment of different consumer and producer groups in the recommendation space [**54**]. Recent works that apply individual fairness notions to GNNs take different perspectives on similarity [**98, 88, 48**]. In particular, *REDRESS* [**48**], one such fairness framework, proposed by Dong et al., frames individual fairness from a ranking perspective. This makes the approach particularly appealing to integration with a downstream task of recommendation.

Our main contribution is the exploration of the applications of this framework to another, more scalable GNN architecture that allows for sub-sampling. In their work, Dong et al. design their framework to be compatible with a Graph Convolutional Network (GCN) [**93**]. However, the message passing paradigm of this method requires training on the entire graph at once which severely limits the applicability of this fairness framework to large scale datasets. As such, our work expands the proposed REDRESS framework to support sub-sampling methods and assesses the effects of this subsampling on the global individual fairness of the final embeddings. We re-implement the methodology proposed by Dong et al. and add our own implementation of a modified GraphSAGE [**71**] which is compatible with individual fairness promotion. Our empirical evaluation on two real world datasets shows that the use of subsampling can drastically improve the individual fairness exhibited in the final embeddings. In doing so, we uncover a connection between neighborhood selection and individual fairness promotion.

3.2. Related Work

In this section we lay the groundwork for the various facets of work which play a crucial role in shaping the contribution and relevance of ours. First, we introduce the role that GNNs play in recommender systems. In doing so, we motivate the relevance of our approach to the recommendation space. Then, we introduce the need for feasible individual fairness notions which can be applied to ameliorate issues of bias in recommender systems. In this discussion, we highlight the ways in which our specific fairness framework, REDRESS [**48**], can be applied to a recommendation setting. We follow this with a discussion of other individual fairness frameworks that have been applied to GNNs. Our discussion contrasts the other approaches with REDRESS, showcasing its intuitive connections with item-item relevance. Finally, we introduce the importance of sub-sampling in GNN mini-batch training. We position our work in relation to other surveys which explore the relation between sub-sampling and utility gains. In doing so, we highlight the crucial contribution of our work: the connection between sub-sampling and individual fairness promotion.

GNNs in Recommender Systems: One of the main factors which cements the success of GNN techniques in the recommendation domain is their ability to capture both context and content based patterns [**163, 165**]. This enables GNNs to encode crucial collaborative signals embedded in a network of users and/or items as well as individual information about items and users themselves. The basis of taste and affinity for an item can be expressed both contextually, via our social circles, and individually, via our interactions with the item itself. As such, GNNs ability to capture both these sources of information enable them to build profoundly complex and robust representations of users and items.

The motivation behind our selection of GraphSAGE [**71**] as a comparison method with the original GCN architecture which is used in the fairness framework proposed by Dong et al. [**48**] is its achievements within the recommender system domain. PinSage [**170**], proposed by the authors of GraphSAGE one year after its initial release, is designed as a downstream extension of GraphSAGE to recommendation tasks. In terms of its architecture, PinSage remains essentially the same as GraphSAGE, with a slight modification to the neighborhood selection method, which uses *importance pooling* to refine the random walks based on the probability of a node's occurrence. After generating item and used embeddings using the iterative representation learning procedure of GraphSAGE, PinSage performs recommendation based on the k-nearest neighbors of an item or a user in the learned embedding space. As such, by implementing fairness in GraphSAGE, we lay the foundations for easily extending this approach to the recommendation task via PinSage.

The Need for Individual Fairness in Recommender Systems:

One of the major issues which plagues the field of recommendation is popularity, or exposure bias [**32**]. This phenomenon occurs when users are only exposed to a portion of the available items. Frequently, this exposure is prioritized in the favor of previously popular items, leaving niche or new items to be neglected in negative feedback loop which catapults some items to *superstar* popularity [**12**] and leaving others languishing in obscurity. This issue has a plethora of negative downstream consequences. First, it deteriorates the overall quality of predictions if a recommender system is able to serve only the needs of a mainstream audience. Second, it can create unfairness among consumers since not all consumption reflect mainstream tastes [**54**]. Finally, it can have very real, extremely negative consequences on the financial prospects of producers [**180**], affecting their ability to create content, services, or merchandise.

Despite a wide range of works approaching exposure and popularity bias from the perspectives of group fairness [**102, 54**] and counterfactual fairness [**103, 176**], the granularity of individual fairness makes the promotion of this notion significantly more difficult. One of the benefits of the REDRESS [**48**], the approach we incorporate into our work, is its flexibility in defining what features/structure is used to define similarity among items (see Section 3.4.3 for more details). For example, in the musical domain, if two songs sound extremely similar then, by selecting the portion of their feature set that defines their sound qualities, we can promote similarity between the two songs and regularize for inconsistencies based on the popularity of their respective primary artists.

Individual Fairness in Graphs: In the broader discussion of fairness in machine learning, the tasks of promoting fairness is often broken into the three stages of a training pipeline: pre-processing (model), in-processing (during training, through regularization, constrained optimization, or novel loss functions), and post-processing (outcome generation). Fairness in graph representation learning follows a similar break down: fairness in the initial graph [**88**], fairness in the mining (or representation building) algorithm [**19, 98, 9, 48**], and fairness in the final representations (or other downstream outcomes) [**88**]. Lastly, the approaches used can also be loosely categorized based on their fairness notions: group [**38, 19, 49, 9**], individual [**48, 98, 88**], and counterfactual [**107**] fairness techniques.

Since the focus of this work is individual fairness, we will now detail some of the most relevant individual fairness notions for GNNs. In their work, Kang et al. convert the premise of a Lipshitz condition into the graph domain [**88**]. They define individual fairness as a trace maximization problem bounded by a fairness tolerance, δ. They break the task of promoting individual fairness into three steps offering three different approaches to satisfy the notion defined above: 1. Pre-processing the input graph 2. In-processing within the model, or 3. Applying post processing on the generated representations. However, due to its reliance on the graph laplacian for defining fairness, unlike our selected method, REDRESS [**48**], the methodology proposed by Kang et al. does not lend itself well to the recommendation setting due to the nature of their fairness definition. Furthermore, in their paper, Dong et al. show that REDRESS outperforms InFoRm in empirical studies [**48**].

Meanwhile, Lahoti et al. take a different understanding of individual fairness [**98**]. Their Pairwise Fair Representation (PFR) model uses a sparse fairness graph with expert defined pairwise similarities to learn fair representations. Similar to the previous method, the methodology proposed by Lahoti et al does not easily lend itself to the recommendation setting. Unlike REDRESS which is able to perform both link prediction and node classification, Pairwise Fair Representations (PFR) is an unsupervised learning method which is unable to perform link prediction.

Sampling Methods in GNNs: Despite the significant advances achieved by the GCN architectures, there are a few drawbacks to this method. The training regime used by this architecture, which performs full-batch training on the entire graph, has two major limitations:

efficiency and scalability. Holding an entire graph in memory whilst performing complex iterative convolutions makes training a GCN on large graphs essentially intractable [**105**]. As such, a series of modifications have been proposed which enable the use of mini-batch training while maintaining the expressive properties of a GCN's convolutional representation building [**71, 170, 80, 34, 182, 36, 173**]. The variety of these sampling methods and their effects on accuracy and performance have been well documented both empirically [**105, 157**] and theoretically [**40**]. As mentioned in our introduction of the REDRESS [**48**] framework, and explored with more detail in Section 3.4, the primary drawback of this method is its reliance on the GCN methodology. By exploring the effects of subsampling with respect to individual fairness, we differentiate our findings from the various surveys that have assessed subsampling through the lens of utility performance.

3.3. Graph Neural Networks

In this section, we review several key preliminary topics which are necessary for the discussion for understanding our discussion of REDRESS in Section 3.4. Advances in the field of GNNs have come in leaps and bounds within the last decade. Here, we briefly review the two methods integral to our work: GCN [**93**] and GraphSAGE [**71**]. For a detailed overview of GNN methods, we refer to the survey paper by Wu et al. [**167**] and a textbook by Hamilton [**70**].

Given graph data, a GNN iteratively collects feature information from the neighbors of a node and integrates this information into the representation of the node. As such, the basic structure of a GNN can be abstracted into two functions, *UPDATE* and *AGGREGATE*. Here, *AGGREGATE* refers to the process of gathering the neighborhood nodes and *UPDATE* refers to the procedure used to update a node's embedding with respect to its neighborhood feature set. Thus, the GNN process of training embeddings, is based on the interactions between these two processes. At each iteration k of training, the *AGGREGATE* function takes the embeddings of u's neighbors, $v \in N(u)$, to generate a message which the *UPDATE* function combines with the previous embedding of node u, $h_u^{(k-1)}$, to generate the updated embedding $h_u^{(k)}$. This process can be formalized as follows:

$$h_u^{(k)} = \mathit{UPDATE}^{(k-1)}\left(h_u^{(k-1)}, \mathit{AGGREGATE}^{(k-1)}\left(\{h_v^{(k-1)}, \forall v \in N(u)\}\right)\right) \tag{3.3.1}$$

More formally, this can be expressed as:

$$h_u^{(k)} = \sigma\left(W_{self}^{(k)} h_u^{(k-1)} + W_{neigh}^{(k)} \sum_{v \in N(u)} h_v^{(k-1)} + b^{(k)}\right) \tag{3.3.2}$$

where $W_{\text{self}}^{(k)}, W_{\text{neigh}}^{(k)} \in \mathbb{R}^{d^{(k)} \times d^{(k-1)}}$ are trainable parameter matrices, $b^{(k)}$ is the bias term, and σ is a non-linear activation function (in our experimentation we use ReLU [**7**]).

3.3.0.1. GCN. The GCN method proposed by Kipf and Welling [**93**] is unique because of its use of neighborhood normalization. Rather than simply aggregating over all the feature vectors of the neighborhood nodes, which can lead to instability, the neighboring feature set is first normalized.

$$h_u^{(k)} = \sigma\left(W^{(k)} \sum_{\{v \in N(u)\} \bigcup \{u\}} \frac{h_v}{\sqrt{|N(u)||N(v)|}}\right) \tag{3.3.3}$$

3.3.0.2. GraphSAGE. GraphSAGE [**71**] makes two major changes to the methodology proposed by GCN: neighborhood selection and aggregation. First, unlike GCN which integrates information from the all the neighbors of a node, GraphSAGE randomly samples p neighbours from the surrounding neighborhood (i.e., $N_p(u)$). Second, unlike GCN, which normalizes based on the degree of the neighborhood nodes, GraphSAGE averages the embeddings of the neighborhood feature set before integrating them with the representation of the current node. As such, this can be expressed as:

$$h_u^{(k)} = \sigma\left(W^{(k)} \cdot \text{MEAN}\left(\{h_u^{(k-1)}\} \bigcup \{h_v^{(k-1)}, \forall v \in N_p(u)\}\right)\right) \tag{3.3.4}$$

where the MEAN aggregator is the elementwise mean over the feature vectors, $h_v^{(k)}$ for all the neighboring nodes, $v \in N(u)$.

3.4. Rank Based Individual Fairness in GNNs

In this section, we review the individual fairness notion called *REDRESS* which is introduced by Dong et al. [**48**]. We then show how REDRESS can be implemented within the end-to-end training paradigm of a GNN.

3.4.1. REDRESS

Our problem definition is couched in the graph setting. We are given a graph, $G = (V, E)$, with a node set, V, and edge set, E. Our setting also involves a set of node-level features, $X \in \Re^{|V| \times d}$ where d is the dimension of each feature vector. Our current task involves training a GNN model, $\mathcal{M}$, to perform the task of link prediction. In this setting, Y and $\hat{Y}$ represent the ground truth and learned node representations. However, this methodology can easily be extended to the recommendation setting by treating the nodes $v \in V$ as items and attempting to predict items which are similar to each other, while maintaining individual fairness among them. By training a model, $\mathcal{M}$ on graph, G, we can learn a series of representations for each node, or item, $\hat{Y} \in \Re^{|V| \times m}$ where m represents the dimension of the final hidden layer in model, $\mathcal{M}$.

The fairness notion used in this work is based on the framework proposed by Dong et al. [**48**]. This fairness notion, which bases itself on ranking, is extremely well suited to recommendation tasks due to its innate interlacing with item to item relevance. We begin by explaining this notion intuitively and follow with a formal definition:

For each node pair u, v in a graph G, we can define their similarity, $s_{u,v}$ based on some similarity metric $s(.,.)$ in which we input their initial feature vector, $X[v] \in \Re^d$ or learned embedding, $\hat{Y}[v] \in \Re^m$. Applying this procedure in a pairwise fashion, we can derive two matrices. The first is termed the *apriori* or oracle similarity, S_G, in which similarity is based on some initial feature based, structural, or expert defined vectors which are independent of the GNN model, M. The second is termed the learned representation similarity, $S_{\hat{Y}}$, in which similarity is based on a learned embedding generated by the training process of a GNN model, M. Using these similarity matrices, we can generate two lists which rank each node with respect to the others. Individual fairness is upheld in this setting if the respecting orderings of nodes in each ranked list is consistent. That is, items which were ordered based on similarity before training, will maintain the same similarity ordering in their embedding space. Formally, we can express this in the following definition:

Definition 3.4.1. (REDRESS: Individual fairness). Given some similarity metric $s(.,.)$, an apriori pairwise similarity matrix, $S_G \in \Re^{|V| \times |V|}$, and a learned pairwise similarity matrix, $S_{\hat{Y}} \in \Re^{|V| \times |V|}$, defined by applying $s(.,.)$ to the learned graph representations, $\hat{Y}$, we say the predictions are individual fair if for each each instance i, the ranked list generated from $S_{\hat{Y}}$ is consistent with the ranked list generated from S_G.

3.4.2. REDRESS Fairness Construction

Minimizing the differences between the two ranked lists can be seen as a form of ranking optimization. In order to formulate this task as a differentiable operation, we can draw on the work in the field of learning to rank [**21**] to formulate loss based on a probabilistic approach. Using these matrices, S_G and $S_{\hat{Y}}$, we define two probability matrices, $P \in \Re^{|V| \times |V| \times |V|}$ and $\hat{P} \in \Re^{|V| \times |V| \times |V|}$. In essence, for every node u_i, we can define its similarity with two neighboring nodes u_j, u_m as $s_{i,j}$ and $s_{i,m}$ in the case of S_G or $\hat{s}_{i,j}$ and $\hat{s}_{i,m}$ in the case of $S_{\hat{Y}}$. Using these values, the matrices P and $\hat{P}$ codify the probability $P_{j,m}(s_{i,j}, s_{i,m})$ that node u_i is more similar to node u_j than u_m.

Thus, for each individual node, u_i and two other nodes selected from the remaining graph, u_j, u_m, the embedding based matrix, $\hat{P}$ is defined as:

$$\hat{P}_{j,m}(\hat{s}_{i,j}, \hat{s}_{i,m}) = \frac{1}{1+\exp\left(-\alpha(\hat{s}_{i,j} - \hat{s}_{i,m})\right)}$$

where α is a scalar which can be treated as a hyperparameter. Similarly, the feature based matrix, P can be defined as:

$$P_{j,m}(s_{i,j},s_{i,m}) = \begin{cases} 1 & s_{i,j} > s_{i,m} \\ 0.5 & s_{i,j} = s_{i,m} \\ 0 & s_{i,j} < s_{i,m} \end{cases}$$

3.4.3. Training Individually Fair Embeddings

In defining these matrices, we can now train for fairness by using standard cross entropy loss. For a node i, we define:

$$L_{j,m}(i) = -P_{j,m} \log \hat{P_{j,m}} - (1 - P_{j,m}) \log(1 - \hat{P_{j,m}}) \tag{3.4.1}$$

And over all the nodes $i \in V$:

$$L_{fairness} = \sum_i \sum_{j,m} L_{j,m}(i) \tag{3.4.2}$$

Finally, in order to ease the computational complexity, we can add a weighing factor based on the changes in NDCG to restrict our modifications to changes in the top k values of each ranked list. Once we define $z_{@k}(.,.)$ as NDCG@k, we can express the fairness loss as:

$$L_{fairness} = \sum_i \sum_{j,m} L_{j,m}(i) |\triangle z_{@k}|_{j,m} \tag{3.4.3}$$

where $\triangle$ represents the change in NDCG between the two lists.

The broad process of training occurs using two stages: utility based loss and utility + fairness based loss. The utility loss, $L_{utility}$, is the classic cross entropy loss:

$$L_{utility} = - \sum_{u,v \in G} Y_{uv} ln \hat{Y}_{uv} \tag{3.4.4}$$

and the joint utility + fairness loss can be defined as:

$$L_{total} = L_{utility} + \gamma L_{fairness} \tag{3.4.5}$$

where γ is a scalable hyperparameter which controls the focus given to fairness. By using the NDCG metric, commonly used in ranking and recommendation tasks, REDRESS formulates a fairness loss that is, intrinsically, performing the relevance/fairness balance that is so important to developing novel bias mitigation techniques in recommendation settings. Furthermore, by creating a loss which can easily interpolate between utility, or relevance, and fairness, REDRESS follows methodology very similar to a large body of work in recommendation fairness [**161, 109**]. .

3.5. Experiments

In this section, we evaluate two implementations of REDRESS, comparing the results between GCN and GraphSAGE as the backbone model, $\mathcal{M}$. This section shows the effectiveness of sub-sampling on utility and fairness through empirical evaluations on two widely known real world benchmark datasets: BlogCatalog [**146**] and Flickr [**147**]. Table 3.1 presents the statistics of these two datasets.
The code and data used to obtain these results can be found at FacctRec2022 Github Repository.

Dataset	# Nodes	# Edges	# Features
BlogCatalog [**146**]	5,196	171,743	8,189
Flickr [**147**]	7,575	239,738	1,406

Table 3.1 – Dataset Statistics

Experiments are implemented in Python using the Pytorch compatible version of DGL [**153**]. In order to remain faithful to the validation splits used in the REDRESS experiments, we split the dataset using 40% of the edges for train, 40% for validation, and 20% for test. For the GCN implementation, we used the hyperparameter settings which had been fine-tuned by Dong et al. During hyperparameter tuning for GraphSAGE, we used grid search to explore learning rates $\sim$ (0.0001, 0.001, and 0.1), negative sampling of (1, 3 and 5) edges per positive edge, dropout of $\sim$ (0.0, 0.01, 0.03), weight decay of either (0.0, 0.001, or 0.003), warmup epochs of (30, 60, 100, 120) and fairness epochs of (30, 60, 100). Finally, to enable valid comparison with the results attained in REDRESS [**48**], we performed PCA with 200 components to lower the dimensionality of the feature sets before assigning them to each node.

The best GraphSAGE results for both BlogCatalog and Flickr are achieved using 2 layers of convolutions, a hidden size of 256, a learning rate of 0.001, no dropout, 30 epochs of "warm up" (utility only) training followed by 60 epochs of utility + fairness training, and a batch size of 32. In both cases, the best neighborhood size performance was achieved using the smallest neighborhood tests. In the case of BlogCatalog, this was a neighborhood of size 5 (per layer) and in the case of Flickr, this was a neighborhood of size 10 (per layer).
Similarity Notions: We use the cosine similarity metric [**91**] to define both the apriori and prediction matrices S_G and $S_{\hat{Y}}$, respectively. The purpose of selecting this method is to combine the idea of individual characteristics native to individual fairness with a commonly used similarity measure within the recommendation domain. More specifically, cosine similarity is a feature based metric which measures the distance between items in their feature or embedding space. Note that, although this method is flexible to selecting only portions of the feature set to define similarity, in our experiments we use the entire feature set when defining cosine similarity between node pairs.

Please note that in order to allow for valid comparison between the results listed in the REDRESS paper and the results which we have achieved, we re-run their implementation with a $\gamma = 1$ for both architectures. As such, the results we are listing here do not reflect the results listed in their paper (since they are using γ values which have been fine-tuned to balance utility and fairness).

Neighborhood Selection: We extensively test various neighborhood sizes of GraphSAGE, motivating the importance of neighborhood selection on fairness promotion. As shown in Table 3.4, we compare neighborhood sizes of 35, 30, 25, 10, and 5 for subsampling of a 2-hop subgraph in GraphSAGE mini-batch training.

Evaluation Metric In order to evaluate the performance of the two models on each dataset, we use the area under receiver operating curve (AUC) metric. Meanwhile, in order to evaluate the fairness performance, we calculate the NDCG@10 [**85**] between the apriori similarities, S_G and learned representation similarities, $S_{\hat{Y}}$ averaged over all the nodes.

3.6. Results

We investigate four research questions in our experiments:

RQ1: How does the selection of sub-sampling technique affect fairness promotion? As shown in Table 3.2, we can see that GraphSAGE not only meets, but exceeds the performance achieved by the GCN method proposed in REDRESS [**48**] even before the bias mitigation loop (see Vanilla GraphSAGE vs REDRESS GraphSAGE). For example, when looking at the experiments run on the Flickr data, the GCN method with REDRESS fairness promotion is able to achieve a maximum fairness of 20.38 using Cosine similarity but GraphSAGE, without mitigation, achieves 30.24. This shows that even the use of sub-sampling is beneficial for individual fairness. Similarly on the BlogCatalog dataset, the REDRESS GCN achieves a maximum fairness of 19.64 using Cosine similarity, while Vanilla GraphSAGE achieves 29.95 and REDRESS GraphSAGE achieves 52.02. Intuitively, we believe that this finding can be attributed to the size of neighborhood which is being used to define individual fairness. In training over an entire graph, we are attempting to fix all the discrepancies in the ranked list between disparate areas of the graph. However, it is possible that the features, or structural elements, affecting fairness have local variations, which are too granular to notice when training over the entire graph. By training over a smaller sub-graph, we are able to use local nuances to fine-tune the embeddings towards individual fairness. See RQ4 for more detailed experiments and explanations.

		Cosine	
Data	Model	AUC	Fairness
BlogCatalog	Vanilla GCN	86.75 (-)	16.60 (-)
	REDRESS GCN	64.15 (-26%)	19.64 (+18%)
	Vanilla GraphSAGE	90.86 (-)	29.95 (-)
	REDRESS GraphSAGE	71.27 (-21%)	52.02 (+73%)
Flickr	Vanilla GCN	86.16 (-)	16.72 (-)
	REDRESS GCN	63.31 (-26%)	20.38 (+21%)
	Vanilla GraphSAGE	85.80 (-)	30.24 (-)
	REDRESS GraphSAGE	64.09 (-25%)	41.51 (+37%)

Table 3.2 – Results for both BlogCatalog and Flickr. Note: the small values in parenthesis indicating a (+/− %) are meant to indicate the change between a utility based model (such as Vanilla GCN) and a fairness promoting model (such as REDRESS GCN)

RQ2: How does the selection of model affect the training time? As shown in Table 3.2, GraphSAGE is able to achieve improved performance on both utility and fairness metrics. In addition, as shown in Table 3.3, the mini-batch training protocol of GraphSAGE enables it to train for significantly less epochs than GCN. For example, on Flickr, GCN required 200 epochs of pre-training (utility based) and 100 epochs of integrated fairness training to achieve it maximal results. Meanwhile, on the same dataset, GraphSAGE required only 30 epochs of pre-training and 60 epochs of fairness training. Similarly for BlogCatalog, GCN required 200 epochs of pretraining and 60 epochs of fairness training, while GraphSAGE required only 30 pretraining epochs and 60 fairness training epochs. For the sake of comparison, we selected minimally sized graphs (less then 100K nodes) to allow for a valid comparison between GCN and GraphSAGE. The intractability of training a GCN on a larger graph has been heavily documented among the graph community [**105, 40**].

		Cosine	
Data	Model	Pre-training epochs	Fairness training epochs
BlogCatalog	GCN	200	60
	GraphSAGE	30	60
Flickr	GCN	200	100
	GraphSAGE	30	60

Table 3.3 – Number of Epochs to Achieve Maximal Fairness Performance

RQ3: How does the fairness promotion affect the utility performance of GraphSAGE? As shown in Table 3.2, we can see that with both GCN and GraphSAGE, the promotion of individual fairness leads to a drop in utility performance. For example, in BlogCatalog, the rounds of cosine similarity based fairness training caused the GCN model's performance to drop by 26% while GraphSAGE dropped by 21%. Given the formulation of the fairness and utility losses, this balance, between utility and fairness, can be managed by tuning the hyperparameter for γ in the loss function (see 3.4.3 for more details). We leave the experimentation of hyperparameter tuning for future work.

RQ4: How does the neighborhood selection affect the ability of GraphSAGE to promote individual fairness? As shown in the discussion of the previous research questions, there is a noticeable improvement in GraphSAGE over GCN for the promotion of individual fairness. We hypothesize that this differences is rooted in the mini-batch training. GraphSAGE's mini-batch training, uses a sampled sub-graph, while GCN uses the entire graph. We believe that the noticeably smaller neighborhood size used in GraphSAGE updates can allow for better fine-tuning of fairness in the representation learning. This is because the features which affect fairness can potentially differ between disparate areas of the graph. In order to test this hypothesis, we perform experiments with 5 neighborhood allocations. As mentioned in 3.3.0.2, GraphSAGE selects a neighborhood by randomly sampling p neighbors from its surrounding neighborhood. As shown in Table 3.4, we can see that as the neighborhood size grows, the fairness performance drops. Thus, showing a clear connection between neighborhood size and fairness promotion.

	Cosine			
REDRESS GraphSAGE	Nodes in layer 1	Nodes in layer 2	AUC	Fairness
	5	5	65.86	58.04
	10	10	46.24	56.95
	25	15	71.27	52.02
	30	30	69.32	48.71
	35	35	67.17	46.29

Table 3.4 – Neighborhood Comparison

3.7. Conclusion

The advances in the field of GNNs have prompted this architecture to become widely adopted into many graph-structured tasks. In tandem with the proliferation of these models, specifically in the space of recommender systems, has come a growing concern of the exponentiation of biases embedded in their outcomes. Although a growing body of recent work has aimed to implement various notions of fairness within the graph space, the area of individual fairness, has not yet been fully explored. Furthermore, many of the methods proposed in this sub-field remain intractable to the large, real world datasets used in recommender settings. Our work sets out to fill in this gap, by implementing REDRESS, an existing framework for individual fairness, and extending it to GraphSAGE, a scalable GNN architecture. In doing so, we show that the addition of mini-batch training via sub-sampling can significantly improve the promotion of individual fairness. Our findings indicate that this improvement is rooted in the neighborhood selection method, which defines the granularity of local fairness patterns. Due to the prevalence of GNNs in recommender systems, these findings also have ripple effects into the domain of recommendations. Given the shown scalability of this individual fairness notion and the significant ties between GraphSAGE and, its down stream recommmender model, PinSage, we lay the foundation for this individual fairness method to be applied to recommendations. By imposing the need for items which are similar in their initial feature space to remain bounded by this similarity in their embedding space, we believe we can mitigate harmful biases prevalent among recommender systems, such as popularity bias. Our findings show the important connection between individual fairness and neighborhood selection. As such, we believe that this work shows the potential for future implications in field of both GNNs and recommender systems.

Chapter 4

Fairness Through Domain Awareness: Mitigating Popularity Bias For Music Discovery

by

Rebecca Salganik[1], Fernando Diaz[2], and Golnoosh Farnadi[3]

([1]) Mila, Université de Montréal

([2]) Carnegie Mellon University

([3]) Mila, Université de Montréal, McGill University

This work has been submitted to a conference at the time of thesis submission and can be publicly accessed at `https://arxiv.org/abs/2308.14601` .

The main contributions of Rebecca Salganik for this articles are presented.

— Proposal and design of research objectives and experimental methodology (under supervision from collaborators);

— Implementation and development of open source code base for reproducing experimental results;

— Writing the paper as first author.

Co-author, **Fernando Diaz**, helped with insightful feedback throughout the project and paper writing.
Co-author, **Golnoosh Farnadi**, guided the project as principal investigator on the project (PI), providing feedback throughout the project and paper writing.

Résumé. À mesure que les plateformes de musique en ligne se développent, les systèmes de recommandation musicale jouent un rôle essentiel en aidant les utilisateurs à naviguer et à découvrir le contenu de leurs vastes bases de données musicales. En contradiction avec cet objectif plus large, il y a la présence d'un biais de popularité, qui amène les systèmes algorithmiques à privilégier le contenu grand public au détriment d'éléments potentiellement plus pertinents, mais de niche. Dans ce travail, nous explorons la relation intrinsèque entre la découverte musicale et les biais de popularité. Pour atténuer ce problème, nous proposons une approche basée sur l'équité individuelle, sensible au domaine, qui corrige le biais de popularité dans les systèmes de recommandation basés sur les réseaux neuronaux graphiques (GNN). Notre approche utilise l'équité individuelle pour refléter une expérience d'écoute de vérité terrain, c'est-à-dire que si deux chansons se ressemblent, cette similitude devrait se refléter dans leurs représentations. Ce faisant, nous facilitons une découverte musicale significative, résistante aux préjugés de popularité et ancrée dans le domaine musical. Nous appliquons notre méthodologie BOOST à deux tâches basées sur la découverte, en effectuant des recommandations à la fois au niveau de la playlist et au niveau de l'utilisateur. Ensuite, nous avons fondé notre évaluation sur le paramètre de démarrage à froid, montrant que notre approche surpasse les références d'équité existantes en termes de performances et de recommandation de contenu moins connu. Enfin, notre analyse explique pourquoi la méthodologie proposée constitue une approche nouvelle et prometteuse pour atténuer les biais de popularité et améliorer la découverte de contenus nouveaux et de niche dans les systèmes de recommandation musicale.
Mots clés : Réseaux de neurones graphiques, équité algorithmique, équité individuelle

Abstract.

As online music platforms grow, music recommender systems play a vital role in helping users navigate and discover content within their vast musical databases. At odds with this larger goal, is the presence of popularity bias, which causes algorithmic systems to favor mainstream content over, potentially more relevant, but niche items. In this work we explore the intrinsic relationship between music discovery and popularity bias. To mitigate this issue we propose a domain-aware, individual fairness-based approach which addresses popularity bias in graph neural network (GNNs) based recommender systems. Our approach uses individual fairness to reflect a ground truth listening experience, i.e., if two songs sound similar, this similarity should be reflected in their representations. In doing so, we facilitate meaningful music discovery that is robust to popularity bias and grounded in the music domain. We apply our BOOST methodology to two discovery based tasks, performing recommendations at both the playlist level and user level. Then, we ground our evaluation in the cold start setting, showing that our approach outperforms existing fairness benchmarks in both performance and recommendation of lesser-known content. Finally, our analysis explains why our proposed methodology is a novel and promising approach to mitigating popularity bias and improving the discovery of new and niche content in music recommender systems.

Keywords: Graph Neural Networks, Algorithmic Fairness, Individual Fairness

4.1. Introduction

The proliferation of market activity on digital platforms has acted as a catalyst for research in recommendation, search, and information retrieval [**78**]. At its core, the goal of this research is to design systems which can facilitate users' exploration of an extensive item catalogue: be it in the domain of journalism [**160**], films [**72**], fashion [**47**], music [**135, 95, 136**], or otherwise. Within this larger goal of recommendation, each domain comes with its own specifics that differentiate it from other settings [**116, 53, 22**]. Particular to the music streaming domain, an extensive body of work has explored the importance of discovery, exploration, and novelty in the larger goal of performing music recommendation [**51, 99, 42, 128, 114, 59**]. Broadly, discovery can be considered the ability of a curatorial system to expose users to relevant content that they would not have manually discovered themselves [**128, 76, 59**]. And, most significantly, a collection of works have shown that music discovery to be the second most important factor for customer loyalty respective to a particular streaming platform due to the gratifying nature of constructing playlists and interacting with an algorithmic curatorial system [**114, 99, 128**].

Crucially, recent work in this domain has begun to uncover an inverse relationship between novelty, one of the keys to discovery, and the notion of popularity bias [**160, 87**]. Within the broader recommendation community, popularity bias has long been an important topic of research. This phenomenon manifest itself when algorithmic reliance on pre-existing data causes new, or less well known items, to be disregarded in favor of previously popular items [**84, 122, 144, 32, 5, 27**]. And, particularly in the context of discovery, where purpose of a user's engagement with algorithmic curation hinges on exposure to musical items which they would not have already been familiar with, the presence of popularity bias can clearly hinder a system's ability to serve this need. In this work, we explore the **intrinsic interplay between discovery and popularity bias** through the lens of graph neural network (GNN) based recommender systems [**165, 58**]. In the graph space, popularity of individual items is deeply interlaced with the degree centrality of a node, or the number of outgoing edges that leave this node and connect it to others in the graph. This is because the innate process of representation learning is affected by the number of neighbors a node has [**89**]. And, thus, a node's centrality can dictate the quality of its learned representation. This suggests that duplicating the feature information of an extremely popular song, creating a new song using these duplicate features, and randomly placing it once at the edge of a graph, will significantly impact its learned representation, even if everything about the song remains *exactly the same*. As we show in our experimentation, one solution to this disparity lies in a debiasing method that is aware of similarities between musical items and is, thus, grounded in the musical domain.

However, current approaches for mitigating popularity bias in recommender systems approach this task in a domain agnostic approach [**132, 3, 108, 174, 159**]. Such abstraction can be extremely relevant to domains in which item and user level features are scarce, sparse, or non existent. However, in an environment like music streaming where there is a plethora of valuable feature information, we believe that grounding fairness notions in domain specific attributes can prove incredibly valuable. In addition, a majority of these methods focus on using either group [**132, 174**] or counterfactual fairness [**178, 180**], often relying on a binary sensitive attribute to encode popularity. This can cause intrinsic limitations because popularity between items is not necessarily a binary state and such attributes may not be readily available.

In this work, we propose a domain aware, individual fairness based approach for facilitating engaging music discovery. In order to facilitate the domain awareness of our approach we generate nuanced multi-modal track features, extensively augmenting two publicly available datasets. Using these novel feature sets, we show the importance of integrating musical similarity into a debiasing technique and show the effects of our method at learning expressive representations of items that are robust to the effects of popularity bias in the graph setting. Grounding our approach in the musical domain empowers us to leverage a ranking-based individual fairness definition and extend it to the bipartite graph setting. In doing so, we design a method that uses music features to fine-tune item representations such that they are reflective of information that is, in essence, a ground truth to the listening experience: two songs that sound similar should, at least somewhat, reflect this similarity in their learned representations. Finally, we compare our individual fairness-based method with three other methods which are grounded in other canonical fairness notions and are not domain-aware. Through a series of empirical results, we show that our fairness framework enables us to outperform a series of accepted fairness benchmarks in both performance and recommendation of lesser known content on two important music recommendation tasks. In summary, the contributions of this paper are the following:

(1) **Problem Setting**: we define the task of music discovery through the lens of domain-aware individual fairness, showing the intrinsic connections between individual fairness, musical similarity, popularity bias, and music discovery.
(2) **Dataset Design**: we extensively augment two classic music recommendation datasets to generate a set of nuanced multi-modal track features, laying the foundation for future domain-aware mitigation techniques.
(3) **Method**: (1) we provide a novel technical formulation of popularity bias (2) design a domain-aware ranking based individual fairness approach to mitigating popularity bias in graph-based recommendation.
(4) **Experiments**: we show that our method outperforms three state of the art fairness benchmarks in the cold start setting.

4.2. Related Work

In this section we contextualize our work by presenting relevant literature on the subjects of (1) popularity bias and (2) graph neural network (GNN) based recommender systems.

4.2.1. Popularity Bias in Recommendation

Most broadly, popularity bias refers to a disparity between the treatment of popular and unpopular items at the hands of a recommender system. As such, this term is loosely tied to a collection of complementary terms including exposure bias [**46**], superstar economics [**12**], long tail recommendation [**108**], the Matthew effect [**112**], and aggregate diversity [**6, 28**]. There have been several different approaches to formulating popularity through some quantitative definition. One body of work defines popularity with respect to individual items' visibility [**176, 46, 108**]. Another group of approaches attempts to simplify this process by applying some form of binning to the raw appearance values. Most notably, the long tail model [**27, 65, 50, 168, 122**] has risen to prominence as a popularity definition. As shown in Figure 4.1, we can see that the first 20% of items, called *short head*, take up a vast majority of the user interactions and the remaining 80%, or *long tail* and *distant tail*, have, even in aggregate, significantly fewer interactions. Often, splitting items into the *short head* and *long tail* (either including or removing *distant tail*) to define disparity in popularity can overcome the issues of range while still representing concrete disparities in item level visibility.

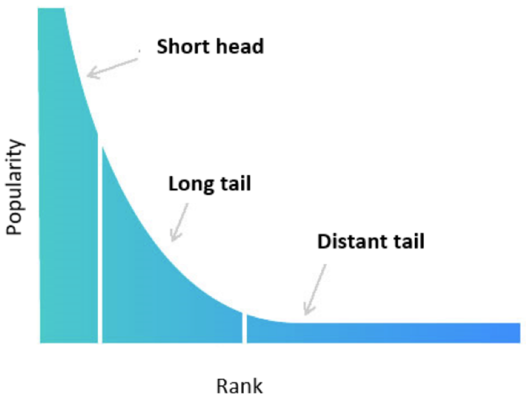

Figure 4.1 – Classic definition of Long Tail popularity settings [**2**]

There has been a lot of work done analyzing and codifying the nature of popularity bias in recommender systems [**23, 83, 84, 32**]. Approaches to mitigating popularity bias are often grounded in one of three methods: **pre-processing** [**17**], **in-processing** [**132, 159, 179**], or **post-processing**, [**3, 108**]. These mitigation strategies are often based on the instrumentation of various canonical fairness notions such as **group fairness** [**17, 138, 132, 174, 3**], **counterfactual fairness** [**159, 179, 176**], or **individual fairness** [**30, 156**].

We contrast our work with previous individual fairness approaches in our use of the music feature space as a form of domain expertise in definition of item-item similarity. We argue that without this "anchoring" an individual fairness method that uses the output of a recommender model, whether it be in learned representation [**156**] or the relevance score [**30**], is already influenced by an item's popularity. Finally, in addition to the classical formulation of popularity bias, a group of works have explored the connection between popularity bias and novelty [**106, 178, 180**] where various metrics are designed to evaluate the novelty of a recommended list. We see our work as complimentary to the exploration in this area however, we differentiate our problem formulation because while novelty is an important aspect of discovery, without domain awareness novelty alone does not account for musical similarity - a critical aspect of the discovery setting.

4.2.2. GNNs in Recommendation

In recent years various graph neural network (GNN) architectures have been proposed for the recommendation domain [**163**]. For brevity, we will focus only on the two methods that are used as the backbone recommenders to the fairness mitigation techniques discussed later in this paper, however we refer to the following surveys [**165, 58**] for recent innovations in this domain.

In particular, *PinSage* [**170**] is an industry solution to graph-based recommendation. Unlike many competing methods, which train on the entire neighborhood set of a node, PinSage trains on a randomly sampled subset of the graph. In order to construct neighborhoods, PinSage uses k random walks to select the top m most relevant neighbors to use as the neighbor set. However, it is important to note that PinSage learns representations of items but not users. Meanwhile, *LightGCN* [**75**], is a method that learns both user and item embeddings simultaneously. Since its proposal in 2020, it is still considered state of the art.

4.3. Methodology

In this section we detail the dataset augmentation procedure and architecture of our domain-aware, individually fair music recommendation system. First, we introduce our datasets in Section 4.3.1. Then, following the problem setting in Section 4.3.2, we reformulate popularity bias in Section 4.3.3 and introduce our domain-aware, individually fair music recommender system in Section 4.3.4.

4.3.1. Dataset Augmentation Procedure

One of the limitations with working on music recommendation is the scarcity of up-to-date, publicly available feature-based datasets. This is because the datasets which are available are often purely interaction-based, meaning that they lack the necessary track-level features to implement domain-aware fairness measures. Thus, one of the preliminary steps of our work was the extensive augmentation of two publicly available datasets: LastFM [**110**] and the Million Playlist Dataset [**31**]. The augmentation and release of these two complementary datasets is an important contribution because it paves the way for further work in domain-aware music recommendation and alleviates the reproducibility issues often posed by the use of music datasets. Although we are limited by the number of publicly available music datasets which are compatible with our feature augmentation procedure, we believe that in selecting these two datasets, we highlight the benefits of our methodology on a broad range of settings related to music recommendation. First, these datasets encompass two important levels of recommendation: playlist (MPD) and user (LFM) based. Second, they showcase two different methods of user feedback data: implicit and explicit. MPD consists of user generated playlists meaning that its interactions consist of songs which are explicitly pronounced as relevant due to the explicit nature of a user selecting the song for their playlist. Meanwhile, LFM contains user/song interactions that are gathered by aggregating all the songs that a user clicked on (even if they did not necessarily finish or enjoy the content). Thus, these implicit interactions have no guarantee of relevance, making the dataset more prone to noise. And, particularly in the cold start setup (see Section 4.4.1), this can significantly increase the difficulty of making predictions because implicit interactions are less indicative of a user's latent taste and less homogeneous in nature than that of a unified playlist.

We augment both of our datasets to include a rich set of features scraped from Spotify API [**1**]. To achieve this, we draw on a large body of work from the music information retrieval community (MIR) [**57, 62**]. We will publicly release the constructed datasets, the construction code, along with the code for using various feature sets, in our repository upon the publication of this paper. The details of the augmented features are as follows.

(1) **Sonic features.** Spotify has a series of 9 proprietary features which are used to define the audio elements associated with a track. They are *danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, and tempo.* Each of these features is a continuous scalar value. In order to normalize the scales, we apply 10 leveled binning to the values.

(2) **Genre features.** We identify the primary artist associated with each collect all the genre tags associated with them. For the emebddings, we select the top 20 and convert them to a one-hot encoding.

(3) **Track Name features.** For each song in the dataset, we extract the song title and pass it through a pre-trained language transformer model, *BERT* [**44**], into an embedding of dimension 512.

(4) **Image features.** For each song in the dataset we extract the associated album artwork. We pass this image through a pre-trained convolutional neural network, *ResNet50* [**74**], to generate an embedding of dimension 1024.

4.3.2. Problem Setting

The task of performing recommendation can be seen as link prediction an undirected bipartite graph. We denote such undirected bipartite graph as $G = (V, E)$. The note set $V = T \cup P$ consists of a set containing song (or track) nodes, T, and playlist (or user) nodes, P (or U). The edge set E are defined between a playlist p_k (or user u_k) and a song t_i if t_i is contained in p_k (or listened to by u_k). Following this setting, our goal (link prediction) is to predict whether any two song nodes $t_i, t_j \in T$ share a common parent playlist p.

4.3.3. Reformulating Popularity Bias

4.3.3.1. Defining Popularity. As mentioned in Section 4.2.1, there is no true consensus within the community on how to define popularity. Here, we present a methodology which we believe allows for both the granularity and expressiveness necessary to highlight differences among various mitigation methods. Broadly, our method consists of important steps (1) logarithmic smoothing and (2) binning. In doing so, we combine the best of each methodology. Applying a logarithmic transformation to the raw values, solves the scaling issues that are caused by the extremes of the long-tail distribution. Meanwhile, binning allows us to provide aggregate statistics that highlight large scale patterns in the recommendations. And,

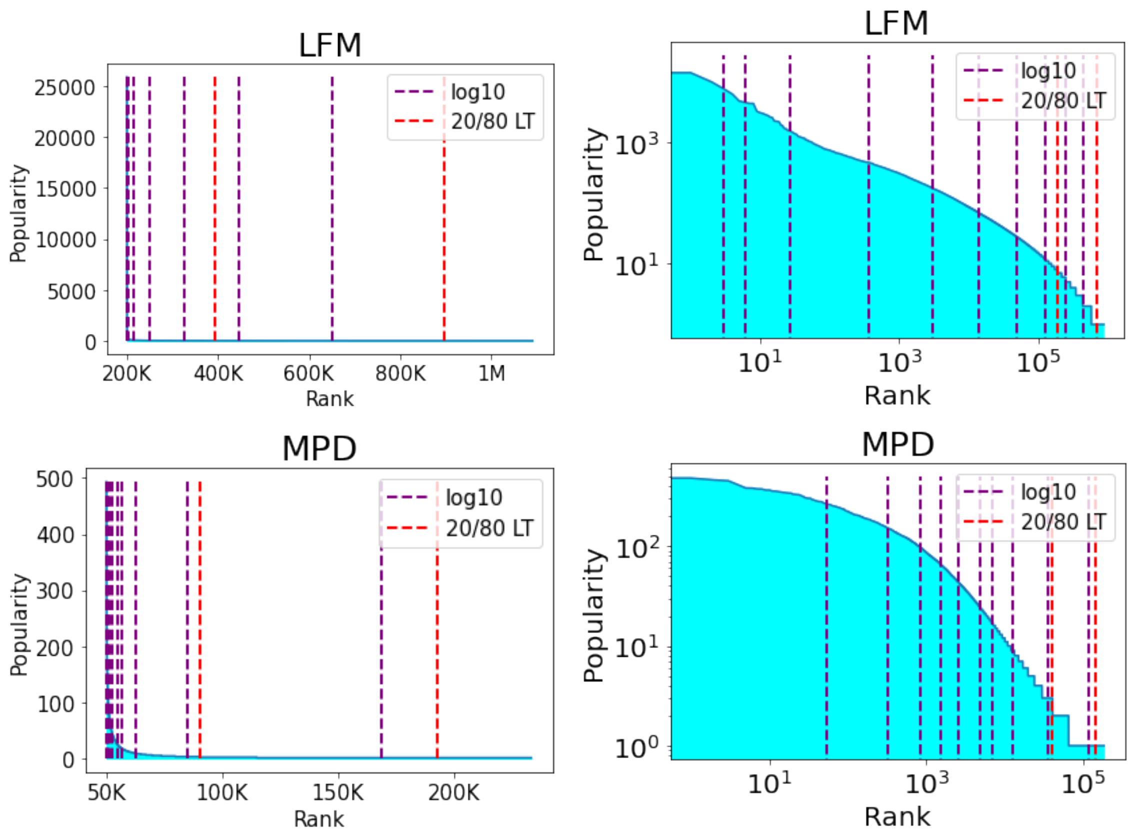

Figure 4.2 – Binning procedure for popularity definition. We show the breakdown of the bin locations for both dataset using our method as compared with the classic long tail model [**108**]. We can see that our logarithmic smoothing and increased bin count allow for a more granular visualization of popularity between various item groups.

while there are many methods which apply binning [**132, 3, 46**], without the logarithmic smoothing, due to the nature of our datasets' distributions the amount of items in the bins would be unevenly distributed, leaving some bins empty. Finally, we select 10 bins based on the distribution of the datasets and the formulation of our BOOST methodology (see Section 4.3.4).

We use the following steps to define our popularity setting. First, we count the number of times each song track, t_i appears within the playlist (or user) training interactions such that for each t_i, $a_{t_i} = |\{p_i : t_i \in p_i\}|$. Then, we apply the base 10 logarithmic smoothing to these values such that for each t_i, $\text{pop}_{t_i} = \log_{10}(a_{t_i})$. Finally, we apply binning onto these values to split them into 10 groups such that for each t_i, $\text{pop\_bin}(t_i) \in \{0, \ldots, 9\}$ where bin 9 has a higher popularity value than bin 0. The visualization of this binning procedure and its comparison with the long tail method can be seen in Figure 4.2. As demonstrated by our visualizations, transforming the raw values into the logarithmic space shows that the bins are filled in relatively even intervals, where, as the popularity increases, so does the number of songs included in a bin.

We showcase the gains that our method has over the canonical long tail model in Figure 4.2 where we compare the positioning of our binning methodology with the classic long tail model. Furthermore, as we later show Figures 4.5 and 4.6 our formulation of popularity is able to elucidate crucial differences among both the datasets and baseline model performances on these datasets.

4.3.3.2. Popularity Bias and Music Discovery. In addition we formalize the inverse relationship between music discovery and popularity bias. For each song track, $t_i \in T_{OG}$, we generate a counterfactual example song, $t_i^* \in T_{CF}$, where everything about the features is *exactly the same* and the only difference is that t_i has a high degree while t_i^* has a degree of one. We calculate the distance between an original song node, t_i and its counterfactual duplicate, t_i^*. A system with high potential for musical discovery will have a low distance between the songs, showing a low popularity bias and an understanding of musical similarity. We will return to this formulation in Section 4.5.1, showing that a node's placement and degree in the graph can exacerbate the presence of popularity bias, reflecting itself in the node's learned representation.

4.3.4. Mitigating Popularity Bias Through Individual Fairness

Ranking-based individual fairness. *REDRESS* is an individual fairness framework proposed by [**48**] for learning fair representations in single node graphs. We extend this framework to the bipartite recommendation setting and integrate it into our popularity bias mitigation approach. In the *REDRESS* setting, individual fairness requires that nodes which were similar in their initial feature space should remain similar in their learned representation embeddings [**52**]. More concretely, for each song node, t_i, and node pair t_u, t_v in a graph G, similarity is defined on the basis of the cosine similarity metric, $s(\cdot, \cdot)$, as applied to either a feature $X[v] \in \Re^d$, or learned embedding set, $Z[v] \in \Re^m$. Applying this procedure in a pairwise fashion produces two similarity matrices. The first, or *apriori similarity*, S_G, in which similarity is calculated on input features and the second, or *learned similarity*, S_Z, in which similarity is calculated between learned embeddings generated by some GNN model, M. The purpose of *REDRESS* is to formulate this as a ranking on the basis of similarity and differentiate on the disparity between differences in rankings of the two representational spaces. Thus, drawing on principles from learn to rank [**20, 21**], each entry in these similarity matrices is re-cast as the probability that node t_i is more similar to node t_u than t_v and transformed into an *apriori probability tensor*, $P_G \in \Re^{|T|\times|T|\times|T|}$, and a *learned probability tensor*, $P_Z \in \Re^{|T|\times|T|\times|T|}$. Having defined these two probability tensors, for each individual node the fairness loss, $L_{t_u,t_v}(t_i)$, can be treated the cross entropy loss applied to these probability distributions such that for an individual node, t_i:

$$L_{t_u,t_v}(t_i) = -P_G[u,v,i]\log P_Z[u,v,i] - (1 - P_G[u,v,i])\log(1 - P_Z[u,v,i]) \quad (4.3.1)$$

and aggregated over all nodes $t_i \in V$ as:

$$L_{\text{fairness}} = \sum_{i}^{|T|}\sum_{u}^{|T|}\sum_{v}^{|T|} L_{t_u,t_v}(t_i) \quad (4.3.2)$$

Individually fair music discovery. The original formulation of individual fairness requires some form of domain expertise [**52**] to determine how (dis-)similar two items are. For the music discovery domain, we use music features (see Section 4.3.1 for exact details) as the basis for calculating cosine similarity. Thus, our *apriori similarity*, S_G, is defined as the cosine similarity between the musical features, $X[v] \in \Re^{|T|\times 9}$, associated with song nodes. Meanwhile, our *learned similarity* contains the song-level embeddings, $S_Z \in \Re^{|T|\times m}$, learned by PinSage. In this way, REDRESS acts as a regularizer that ensures that rank-based similarity between songs is preserved between the input and embedding space. Thus, our similarity notion is domain-aware and grounded in the essence of musical experiences: acoustics.

To learn the embedding of songs, we follow the learning paradigm of PinSage [**170**] and make a few deviations. Unlike [**170**], we use uniform random sampling to avoid the computational burden of calculating negative samples on our large graph and compensate for the potential loss of information by using focal loss [**104**], rather than the margin loss, to train the network. It is important to note that since the potential benefits or drawbacks of PinSage as a general recommender system are out of the scope of this paper, we do not focus on the performance gains that such a change might provide and leave the addition of various negative sampling techniques to future work.

Bringing popularity into individual fairness. The *REDRESS* framework does not explicitly encode any attributes of popularity in its training regimen. Thus increased visibility given of niche items comes only from innate similarities in the musical features, not explicit promotion of niche content. To extend this technique for explicitly counteracting popularity bias, we define the BOOST technique which is used to further increase the penalty on misrepresentation of items that come from diverse popularity categories. As mentioned in Section 4.3.3, we define 10 popularity bins by applying a logarithmic transformation and binning the degrees of a node i (i.e., deg_i) such that $\text{pop\_bin}(i) = \text{bin}(\log_{10}(deg_i))$. This popularity bin can then be integrated with the REDRESS calculations. More formally, given the learned representation matrix, $S_Z \in \Re^{|T|\times|T|}$, we define another matrix B in which

$$B_{ij} = |\text{pop\_bin}(i) - \text{pop\_bin}(j)| \quad (4.3.3)$$

Then, in the *BOOST* loss formulation, in place of S_Z we use $S'_Z = S_Z + B$.

Objective function. The representation learning objective used during training is:

$$L_{\text{total}} = L_{\text{utility}} + \gamma L_{\text{fairness}} \tag{4.3.4}$$

where γ is a scalable hyperparameter which controls the focus given to fairness and can be used to select a balance between utility (L_{utility}) and fairness (L_{fairness}) during the second training phase. For L_{utility}, we apply the aforementioned focal loss [**104**]. And L_{fairness} is Equation 4.3.2 defined above.

Generating recommendations. Crucially, the PinSage architecture is only designed to learn embeddings for songs, not for playlists (or users). Thus we design our own procedure for generating playlist (or user) embeddings using the learned song embeddings. For each playlist (or user) node, p_i, we have a set of songs, $T(p_i) = \{t_i \in T, e_{p_i,t_i} \in E\}$, which are contained in a playlist. For a test playlist, p_t, we split the associated track set into two groups:

$$T(t_p) = \{t_i : t_i \in u_i\} = t_{peek} \cup t_{holdout}$$

such that t_{peek} is a set of k songs that are used to generate the playlist representation and $t_{holdout}$ are masked for evaluation. Thus, in order to generate a playlist (or user) embedding we define:

$$z_{p_t} = MEAN(\{z_{t_j} : t_j \in t_{peek}\})$$

where the $z_k \in \Re^{1\times d}$ are the learned representations of dimension d. Having learned these playlist representations, we apply cosine similarity between an individual playlist, z_{p_t}, and the set of songs in the database, $Z_T = \{z_{t_j} : t_j \in T\}$, selecting the top-k items by their score. We leave further experimentation on designing user-based embeddings via the PinSage paradigm for future work.

4.4. Experimental Settings

In this section we introduce the experimental settings, defining the recommendation scenario (Section 4.4.1), datasets (Section 4.4.2), evaluation metrics (Section 4.4.3), baselines (Section 4.4.4) , and reproducibility settings (Section 4.4.5) encompassed in our experimentation.

4.4.1. Recommendation Scenario

As user consumption habits have shifted away from albums and towards playlists, streaming companies have invested significant energy into the task of *Automatic Playlist Continuation* and *Weekly Discovery* [**137, 143**]. In the first task, *Automatic Playlist Continuation* requires the recommender system to perform *next k recommendation* on a user generated playlist. Meanwhile, in *Weekly Discovery*, rather than augmenting a specific playlist, an

Table 4.1 – Dataset statistics

Dataset	Recommendation Setting	Feedback Type	#Users/Playlists	#Songs	#Artists
MPD	*Automatic Playlist Continuation*	Explicit	11,100	183,408	37,509
LFM	*Weekly Discovery*	Implicit	10,267	890,568	100,638

algorithm is tasked with the creation of a new playlist based on a user's aggregated listening habits. Given our similar treatment of the playlists and users in this recommendation setting, we will use them interchangeably in our formal definition of the task.

Definition 4.4.1. Automatic Playlist Continuation/Weekly Discovery: Given a set of users U, or playlists, $P = P_{train} \cup P_{valid} \cup P_{test}$, and a set of songs (or tracks) T, our goal is to generate a set of k recommendations, $R(P_{test})$.

Following the paradigm of the cold start setting [**137**], we extract train/validation/testing splits on the playlist level by randomly sampling without replacement such that each split trains on a distinct subset of the playlist pool. In this way, we simulate the real world situation in which new users are joining the platform and require relevant, unbiased recommendations without providing a large body of their previous interaction data. It is exactly at this junction, before a user's musical preference solidifies, that the need to mitigate popularity bias is most acute because once a majoritarian pattern has been installed in a user's embedding, it will continue to influence all further music discovery.

4.4.2. Datasets

As introduced in Section 4.3.1, we extensively augment two publicly available datasets, LastFM (LFM) [**110**] and the Million Playlist Dataset (MPD) [**31**], with rich multi-modal track-level feature sets. Table 4.1 presents the graph statistics of both datasets.

4.4.3. Evaluation Metrics

In addition to canonical utility metrics, we design a series of metrics to analyze the effectiveness of our debiasing methods from both a musical and fairness perspective (see Table 4.2 for the details of the formulations).

4.4.3.1. Music Performance Metrics. The purpose of these metrics is to broaden the scope of evaluation to include hidden positive hits. We use *Artist Recall* to evaluate a system's ability to identify correct artists in a recommendation pool, an auxiliary task in music recommendation [**95**]. In addition, we design *Sound Homogeneity* to capture the musical cohesiveness of the recommended songs in a playlist [**16**].

4.4.3.2. Fairness Metrics. To assess the debiasing techniques used to promote of long tail songs we draw on a series of metrics which have been previously used to evaluate the fairness performance of a model [**3, 123, 27**]. Percentage metrics capture the ratio of niche to popular

Table 4.2 – Music and fairness performance metrics. We define a ground truth set, G, and a recommended set, R, we define the set of unique artists in a playlist as $A(.)$ and the d-dimensional musical feature matrix associated with the tracks of a playlist as $F(.) \in \Re^{|\cdot| \times d}$.

Metric	Category	Formulation
Artist Recall@100	Music	$\frac{1}{\lvert P_{test}\rvert} \sum_{p \in P_{test}} \frac{1}{\lvert A(G_p)\rvert} \lvert A(G_p) \cap A(R_p)\rvert$
Sound Homogeneity@100	Music	$\frac{1}{\lvert P_{test}\rvert} \sum_{p \in P_{test}} \cos(F(t_i), F(t_i))\ \forall (t_i, t_j) \in R_p$
Artist Diversity (per playlist)	Fairness	$\frac{1}{\lvert P_{test}\rvert} \sum_{p \in P_{test}} \frac{1}{\lvert A(P)\rvert} \lvert\{A(R_p)\}\rvert$
Percentage of Long Tail Items	Fairness	$\frac{1}{\lvert P_{test}\rvert} \sum_{p \in P_{test}} \frac{1}{\lvert p\rvert} \lvert\{t_i : t_i \in R_p \cap t_i \in LT\}\rvert$
Coverage over Long Tail Items	Fairness	$\frac{1}{\lvert LT\rvert} \lvert\{t_i : t_i \in R \bigcap t_i \in \lvert LT\rvert\}$
Coverage over Artists	Fairness	$\frac{1}{\lvert A\rvert} \lvert\{arid(t_i) : t_i \in R\rvert\}$

content that is being recommended on a playlist (or user) level. Meanwhile, *coverage* looks at the aggregate sets of niche songs and artists over all recommendations. If a recommender has a high percentage but low coverage, the same niche items are being selected many times. Meanwhile, if an item has high coverage but a low percentage, the algorithm is selecting a diverse set of niche content but recommending it very rarely. The gold standard is a high value on both metrics.

4.4.4. Baselines

First, we use two naive baselines: **(1) Features**: Instead of the learned representations, we use the raw feature vectors and **(2) MostPop**: we calculate most popular tracks in each dataset and recommend them each time. Then, we select three state of the art fairness mitigation techniques: **(1) ZeroSum**[**132**]: an in-processing group fairness that defines a regularization term which forces scores within negative and positive item groups to remain close. Following their original implementation, we select LGCN [**75**] as the backbone recommender. **(2) MACR**[**159**]: an inprocessing method which uses counterfactual estimation to denoise for the effects of popularity bias in user and item embeddings. Here too, Following their original implementation, we select LGCN [**75**] as the backbone recommender, and **(3) Smooth xQuAD**[**3**]: a post-processing method that reranks recommendations to improved diversity.

4.4.5. Parameter Settings & Reproducibility

Each of the baseline methods was tested with learning rates $\sim (0.01, 0.0001)$, embedding sizes of $[10, 24, 64, 128]$ and batch sizes of $[256, 512, 1024]$. For the values in the tables below, each stochastic method was run 5 times and averaged. All details and further hyperparameter settings can be found on our GitHub repository [1].

4.5. Results

In this section we present the results of our experimentation. First, we show the connections between individual fairness, popularity bias, and music discovery in the graph domain. Then, we evaluate our method, comparing with a series of the debiasing benchmarks.

4.5.1. RQ1: How does incorporating individual fairness improve the mitigation of popularity bias and facilitate music discovery?

To showcase the performance of our algorithm in the discovery setting and motivate the need for individual fairness in the mitigation of popularity bias, we draw on the definition of music discovery presented in Section 4.3.3.2 by evaluating the effects of popularity bias on learned representations of popular and unpopular songs.

To simulate a situation of maximal popularity bias, we consider the hypothetical example in which extremely popular songs are reversed to become unpopular and measure the effects of degree on their learned representations. From a discovery perspective, the purpose of this simulation is to imagine the most popular song by a listener's favorite artist before it became popular. Our simulation aims to approximate how likely it is that they have discovered the song in relation to its musical attributes, with and without debiasing for the effects of popularity. More formally, for each song track, $t_i \in T_{OG}$, we generate a counterfactual example song, $t_i^* \in T_{CF}$, where everything about the features is *exactly the same* and the only difference is that t_i appears in many playlists while t_i^* appears only once. We augment the original dataset to include these counterfactual songs, $T = T_{OG} \bigcup T_{CF}$. Then, we use five methods to learn the item level representations: one baseline recommender, PinSage, and four bias mitigation methods, ZeroSum [**132**], MACR [**159**], REDRESS, and BOOST. We apply 2-dimensional PCA to each embedding set and analyze the Euclidean distance between the centroids of original track embeddings, $\bar{T}_{OG}$, and counterfactual track embeddings, $\bar{T}_{CF}$. Due to the size of our dataset, we run this metric using the 100 most popular tracks in the MPD dataset and leave further exploration of this phenomenon for future work.

1. preliminary version: `https://anonymous.4open.science/r/RecSys23-9B7F/README.md`

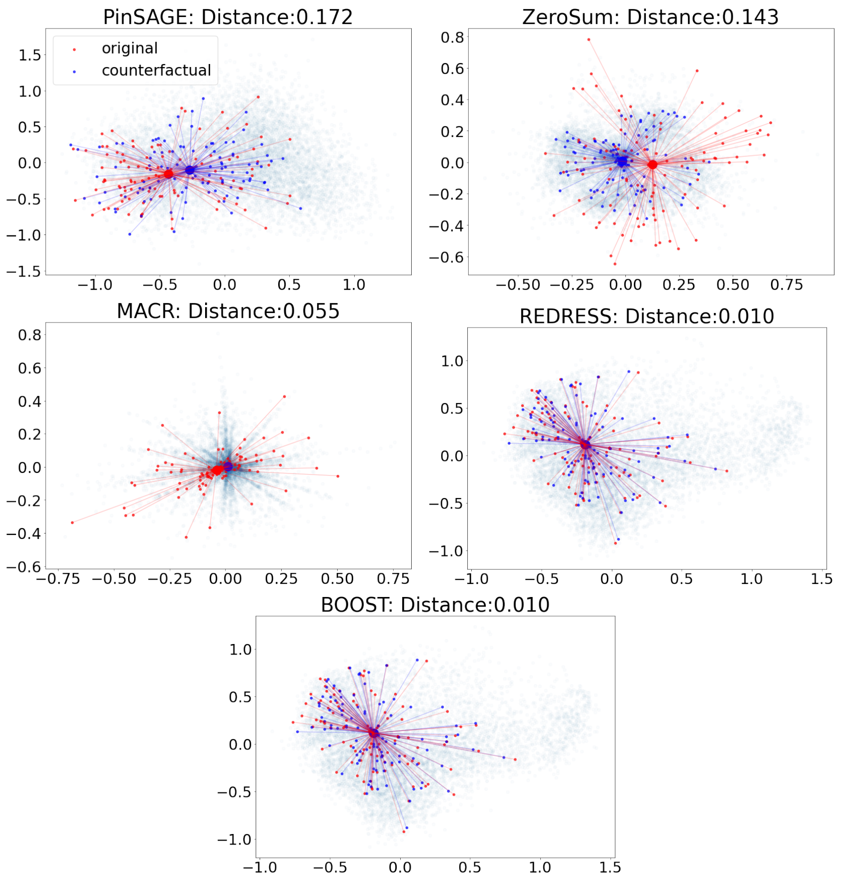

Figure 4.3 – Simulating Popularity Bias: We select 100 of the most popular songs in MPD [**137**], duplicate their node level features, add them with new song track ids, and connecting them to one randomly selected playlist. Then, we analyze the distances between the embedding group centroids. We find that REDRESS and BOOST have the lowest distance between the original, popular and duplicated, unpopular track groups, showing the least amount of popularity bias.

As shown in Figure 4.3, we find that all fairness interventions decrease the distance between the two centroids. Furthermore, as the granularity of fairness increases, the distance between the centroids of learned representations decreases. For example, PinSage, which has no mitigation of popularity bias, has the largest distance of 0.172. ZeroSum [**132**], which considers group fairness, decreases the distance to 0.143, MACR [**159**], which uses counterfactual estimation, shrinks to 0.055. Finally, our methods, REDRESS and BOOST are able to achieve both the lowest distance and the correct orientation between the two embedding spaces.

In these results, we see that the domain-awareness of our methodology, which enables it to understand musical similarity between items, allows it to be robust to the effects of popularity bias on a learned song embedding. Thus, in the setting of musical discovery, it is able to uncover proximity between items which are musically coherent even if they are not

Table 4.3 – Comparison between all methods.Note: We use bold highlights to represent the best performance within a column for each of the datasets. The p values of this table are calculated by applying the Wilcoxon signed-rank test [**131**] to results between PinSage and BOOST. As you can see, the *BOOST* method achieves the best performance along all Fairness metrics when compared with the other debiasing benchmarks.

		Classic		**Music**		**Fairness**			
Data	**Model**	Recall@100	NDCG@100	Artist Recall@100	Flow	Diversity	%LT	LT Cvg	Artist Cvg
MPD	**Features**	0.041	0.073	0.073	0.900	0.841	**0.588**	0.022	0.073
	MostPop	0.044	0.048	0.141	0.908	0.680	0.0	0.0	0.001
	LightGCN	**0.106 ± 0.004**	0.119 ± 0.004	**0.272 ± 0.011**	0.905 ± 0.000	0.672 ± 0.025	0.002 ± 0.000	0.000 ± 0.000	0.025 ± 0.001
	PinSage	0.068 ± 0.002	**0.144 ± 0.003**	0.139 ± 0.003	0.931 ± 0.001	0.707 ± 0.003	0.476 ± 0.002	0.032 ± 0.000	0.105 ± 0.000
	ZeroSum	0.044 ± 0.002	0.043 ± 0.002	0.220 ± 0.008	0.904 ± 0.001	0.765 ± 0.013	0.000 ± 0.003	0.000 ± 0.000	0.048 ± 0.002
	xQuAD	0.064 ± 0.005	0.104 ± 0.006	0.135 ± 0.013	0.927 ± 0.004	0.703 ± 0.059	0.226 ± 0.001	0.017 ± 0.000	0.098 ± 0.004
	MACR	0.028 ± 0.014	0.030 ± 0.015	0.149 ± 0.022	0.902 ± 0.002	0.831 ± 0.034	0.019 ± 0.006	0.000 ± 0.001	0.011 ± 0.003
	REDRESS	0.045 ± 0.002	0.100 ± 0.003	0.162 ± 0.004	0.969 ± 0.032	0.829 ± 0.001	0.504 ± 0.003	0.036 ± 0.004	0.117 ± 0.000
	BOOST	0.020 ± 0.004	0.047 ± 0.003	0.137 ± 0.002	**0.979 ± 0.000**	**0.899 ± 0.002**	**0.522 ± 0.001**	**0.037 ±0.003**	**0.125 ±0.000**
	p values	4.408083e-16	1.768725e-19	0.727897	3.751961e-61	1.168816e-29	0.000596	-	-
LFM	**Features**	0.033	0.037	0.041	0.996	0.919	0.486	0.005	0.034
	MostPop	0.015	0.011	0.046	0.926	0.600	0.000	0.000	0.001
	LightGCN	0.026 ± 0.001	0.023 ± 0.001	0.068 ± 0.001	0.998± 0.000	0.505 ± 0.012	0.000 ± 0.000	0.000 ± 0.000	0.003 ± 0.001
	PinSage	**0.064 ± 0.001**	**0.095 ± 0.002**	**0.077 ± 0.002**	0.969 ± 0.000	0.775 ± 0.003	0.437 ± 0.001	0.008 ± 0.000	0.053 ± 0.001
	ZeroSum	0.001 ± 0.003	0.001 ± 0.001	0.045 ± 0.004	0.996 ± 0.008	0.866 ± 0.000	0.007 ± 0.000	0.000 ± 0.000	0.032 ± 0.001
	xQuAD	0.055 ± 0.001	0.064 ± 0.001	0.068 ± 0.002	0.998 ± 0.000	0.801 ± 0.008	0.212 ± 0.000	0.004 ± 0.000	0.053 ± 0.001
	MACR	0.014 ± 0.001	0.014 ± 0.001	0.049 ± 0.007	0.996 ± 0.003	0.777 ± 0.050	0.002 ± 0.004	0.000 ± 0.000	0.001 ± 0.000
	REDRESS	0.038 ± 0.002	0.053 ± 0.004	0.057 ± 0.001	0.998 ± 0.002	0.862 ± 0.004	0.451 ± 0.000	0.008 ± 0.002	0.056 ± 0.000
	BOOST	0.005 ± 0.001	0.007 ± 0.001	0.029 ± 0.002	**0.999 ± 0.000**	**0.941 ± 0.003**	**0.498 ± 0.006**	**0.010 ± 0.000**	**0.068 ± 0.001**
	p values	5.696989e-08	1.179627e-15	1.914129e-07	0.001408	1.112495e-34	2.477700e-11	-	-

necessarily of similar popularity status. And, in doing so, we build representations that are complex, expressive, and effective for music recommendation.

4.5.2. RQ2: How does our individual fairness approach compare to existing methods for mitigating popularity bias?

In Table 4.3 we show a side by side comparison of the various recommendation and debiasing methods. We apply the Wilcoxon signed-rank test [**131**] to the BOOST, PinSage results to assess the statistical significance of our method's performance. In Table 4.3, we select only the best hyperparameter results for each method. However, for each fairness baseline, there is a hyperparameter which tunes the balance between fairness intervention and performance, γ. Thus, we present Figure 4.4 to show the ranges of these hyperparameter values.

Analyzing Debiasing Performance: First, we look at the comparison between the backbone recommender systems and their debiasing counterparts. Within the greater fairness community it is typical to see a trade-off between recommendation utility and the effectiveness of a debiasing technique [**76**]. And, indeed, in our experiments we witness such a trade-off. For example, evaluating the columns of *Recall* and *NDCG* on Table 4.3 we can see that both recommender systems outperform their debiasing counterparts. This can be largely attributed to the formulations of the utility metrics, the interaction with the debiasing objectives, and the underlying distribution of the datasets. Since the premise of the canonical recommendation utility metrics is to reward a system that can accurately recover the exact tracks a user liked, any attempts to promote long tail content that wasn't originally listened to is penalized, even if it isn't truly indicative of a user's underlying taste. We note that the trade-off is significantly more severe in the LFM dataset as opposed to the MPD dataset. As shown in Figure 4.5, this can be attributed to the underlying distribution of a datasets. Where MPD has a training set which contains a significant portion of interactions on the lower end of the popularity spectrum, LFM skews towards higher popularity. Thus, if evaluating using utility metrics that penalize mistakes on the track-level prediction, even if a system is selecting musically coherent and relevant content, this trade-off becomes inevitable. Indeed, within the music recommendation community, there have been several works suggesting that this trade-off, though present in offline testing, doesn't necessarily carry over into online testing [**26, 76**]. Thus, to provide a deeper analysis of our debiasing methodology, we present two musically relevant metrics, *Artist Recall* and *Flow*. As explained in Section 4.4.3 and detailed in Table 4.2, the *Artist Recall* metric evaluates the recommender systems ability to identify correct artist-level recommendations and the *Flow* evaluates the overall homogeneity of the selected music. In particular, *Flow* plays an important role in the music discovery task because studies have indicated that users are drawn to homogeneous listening suggestions when engaging with algorithmic curation [**16, 76**]. As we can see in both datasets, REDRESS and BOOST consistently achieve the highest *Flow*. This is because, by harnessing musical features and in our debiasing technique, our method generates representations that are indicative of musical similarity, which affects the downstream musical similarity of the recommendations it generates. Meanwhile, looking at the *Artist Recall* columns, we can see a much less significant drop (or, in the case of MPD an increase) in the performance between backbone recommender models and their debiasing counterparts. Crucially, if we consider the implications of such a debiasing technique on a user who's taste skews towards popular music, high performance on these metrics means that our debiasing methods' awareness of musical similarity will enable it to maintain the stylistic elements that a user is drawn to while simultaneously promoting niche content.

Next, we compare the performance among the various fairness promotion methods. Looking at the columns of *recall* and *ndcg* on Table 4.3, we can see that, as expected, xQuAD [**3**] which is a re-ranking method is able to preserve the highest utility. However, we note that among the remaining methods, REDRESS is able to achieve the second highest utility. Meanwhile, if we look at the fairness metrics, we can see that REDRESS and BOOST are the highest performing methods. In particular, looking at the columns for *%LT* and *LT Cvg*, we can see that REDRESS is noticeably better than the other methods and BOOST is able to improve on its performance. Crucially, our method is able to have high values in both coverage and percentage of long tail items meaning that REDRESS/BOOST is not just prioritizing niche items but also choosing a diverse selection from this category. We attribute the relative gains of REDRESS and BOOST to their ability to integrate musical features into their fairness mitigation because they are able to select not just niche items,

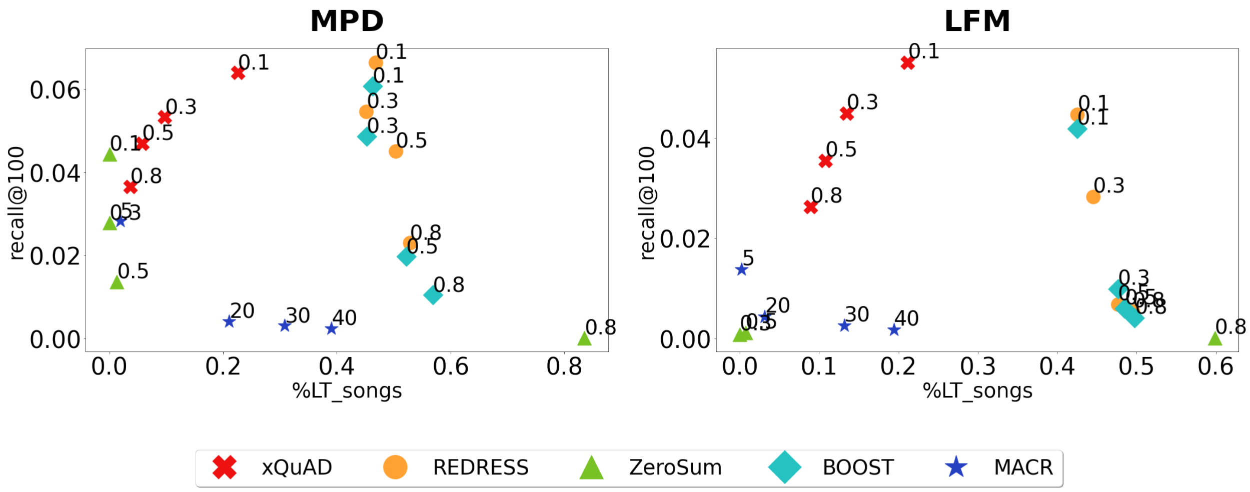

Figure 4.4 – Plots showing the effects of fairness intervention method on performance/fairness metrics: we show the effects of tuning a hyperparameter to balance fairness and performance in each of our fairness methods, we explore the entire range of the values and report the trade-off that increasing this fairness intervention can have. Note, that REDRESS has the best robustness with respect to balancing recall and %LT.

but also musically relevant ones for recommendation. Finally, our method's ability to interpolate between these two perspectives of content and consumption patterns, shows that REDRESS/BOOST is able to recommend similar ratios of niche items compared to the bare features while having significantly better performance.

Hyperparameter Sensitivity: While the results in Table 4.3 are compared among the best hyperparameter tuning that balances between utility and fairness, we also present Figure 4.4 where we show the balance between *%LT* and *recall* along the range of each method's hyperparameter value. For example, xQuAD, ZeroSum, REDRESS and BOOST all have ranges that scale between $(0.1, 0.9)$ and MACR has a value somewhere along $(0, 45)$. As we can see, for any value of hyperparameter along the various methods, REDRESS and BOOST are able to outperform the collection of benchmarks. Given these results, we conclude that our BOOST approach is able to achieve the most effective debiasing performance while REDRESS is able to achieve the most balanced performance.

Popularity Definition: As we can see in Figure 4.6 the definition of popularity plays a significant role in the model selection method *especially* in the case where user preferences encoded in the training data skew towards popular items. In particular, using a less granular definition for popularity bins can synthetically inflate the performance of $\%LT$ and $LTCvg$. For example, we can see that methods like xQuAD and ZeroSum are selecting a majority of their items from bins 1,2 or 3. Using a classical long tail methodology, these differences would not be as visible, masking distinctions among the baselines' fairness.

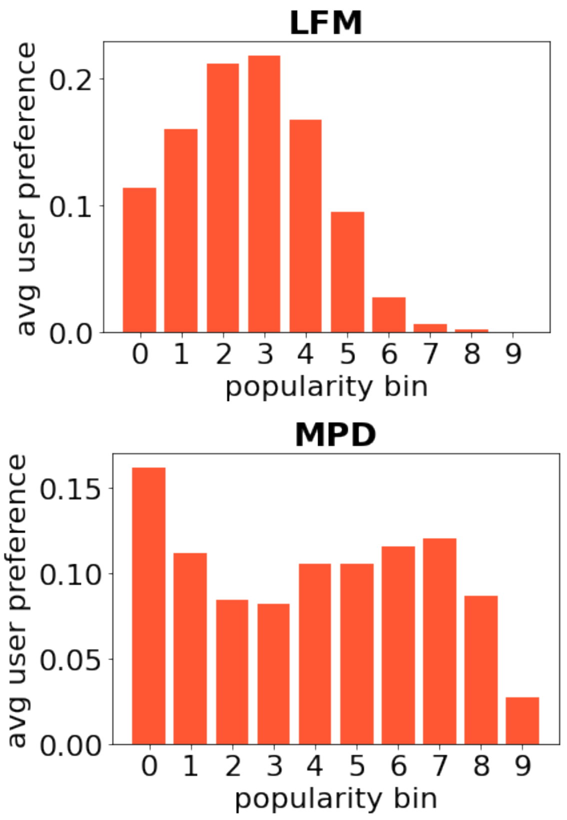

Figure 4.5 – Dataset Breakdown by Long Tail Definition: We show visualizations of the user preferences indicated in the training set for each of the datasets used in evaluation. Using our formulation of popularity we can see that the two datasets have different distributions of popularity in their training data which, in turn, helps explain fairness/performance tradeoffs.

4.6. Conclusion

In this work, we address the problem of mitigating popularity bias in music recommendation. Starting from the perspective of discovery and how it relates to algorithmic curation, we consider the effects of popularity bias on users' ability to discover novel and relevant music. On the basis of this motivation, we highlight the intrinsic ties between popularity bias and individual fairness on both song and artist levels. We ground our individual fairness notion in the music domain, presenting a method to mitigate popularity bias through fine tuning of representation learning via musical similarities. We perform extensive evaluation on two music datasets showing the improvements of our domain aware method in comparison with three state of the art popularity bias mitigation techniques. We hope that these promising findings showcase the importance of developing domain aware methods of mitigating popularity bias in addition to domain agnostic options.

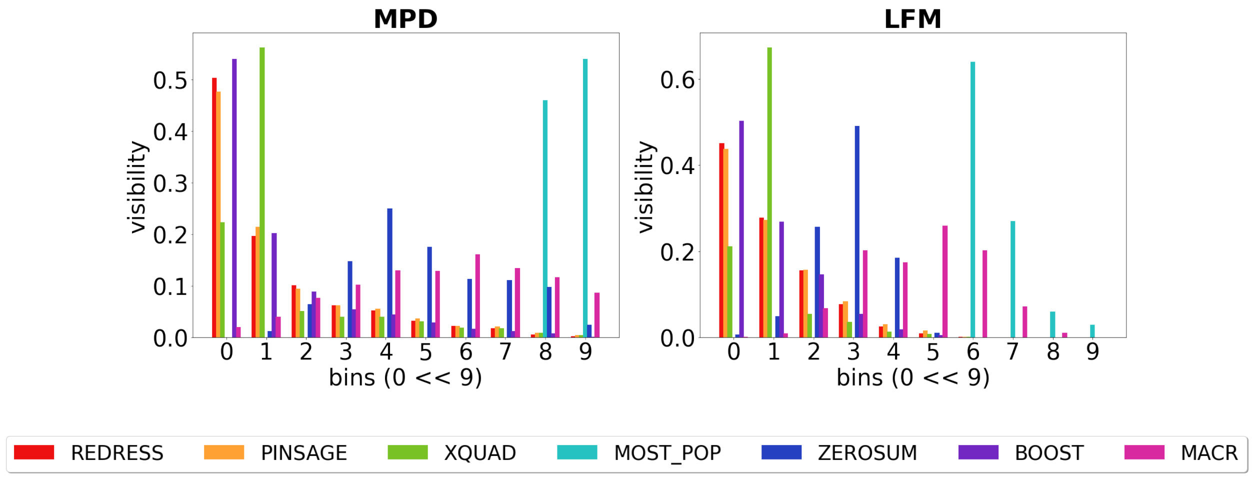

Figure 4.6 – Group By Group Analysis of Recommendations: we look at a breakdown of the recommendations for each dataset. We define visibility as the number of times an item from this group appears in the recommendations normalized by the total number of items in the recommendation lists. Bins are defined using the methodology of Section 4.3.3 where bin 0 has the lowest popularity.

Chapter 5

Conclusion

The focus of this work has been the exploration of domain-aware strategies for mitigating popularity bias in music recommendation. We began by defining the relationship between popularity bias and individual fairness, cementing the intrinsic connection between these two important research directions through the unique needs and objectives of music discovery. Then, on the basis of this interconnected relationship, we presented a debiasing methodology for combating popularity bias in music recommendation that is grounded our individual fairness notion in the music domain. Finally, we showed the effectiveness of our approach in mitigating popularity bias in graph based recommendation through fine tuning of representation learning via musical similarities. Ultimately, the goal of this work is to highlight the potential for domain-aware approaches to mitigating popularity bias in music recommendation. We hope that the work of this thesis presents a compelling argument why grounding fairness approaches in musical notions is an important research direction. As algorithmic curation becomes a mainstay in facilitating music recommendation for millions of listeners all over the world, it is up to the researchers designing these systems to consider the larger implications of their architectural choices on every facet of the music creation process.

5.1. Limitations of Our Work

It is important to note that there are several limitations of the work presented in earlier sections. First, due to the limited nature of publicly available datasets, we lack access to the underlying mechanisms used by Spotify to generate the various modalities of features used in our dataset augmentation. We suspect that both of the datasets used in our experimentation may skew towards Western, anglophone content and would not be representative of the wide array of music that is available for consumption. As such, future work on understanding how the various feature extraction techniques which were used to achieve the metadata used in our analysis can affect the downstream fairness among tracks is an important avenue for further exploration. Second, due to the lack of access to online evaluation mechanism, we must acknowledge that our understanding of both relevance and quality in music recommendation presupposes the effectiveness of various offline metrics (such as recall or ndcg) as proxies for true user engagement. Furthermore, it is important to remember that recommender systems are responsible for serving the tastes of listeners, not policing them, and we do not deny the validity of mainstream listening practices. Given our focus on the mitigation of bias, we cannot say how well our method serves the tastes of different listener profiles and whether our mitigation technique might affect those with majoritarian tastes. We leave this deeper user preference analysis for future work.

5.2. Future Work

There are several directions for future extensions and improvements to the work presented before. For example, the work in Chapter 3 has uncovered an underlying relationship between various message passing paradigms and individual fairness in the learned representations of nodes. We believe that understanding the roots of this surprising finding could be instrumental in the design of future fair and effective graph neural network structures. Additionally, in Chapter 4, we focused on designing a debiasing technique that is compatible with the representation learning paradigm of GraphSAGE and later PinSage. While grounding our debiasing technique in a well-known GNN recommender system provides a context for evaluating its results, designing a recommender system that is centered around provide node-level individual fairness without requiring additional mitigation methodologies as an important and necessary future direction of research.

Bibliography

[1] Spotipy: Spotify api in python, 2014.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. Controlling popularity bias in learning-to-rank recommendation. In *RecSys '17*. ACM, 2017.

[3] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. Managing popularity bias in recommender systems with personalized re-ranking, 01 2019.

[4] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. Addressing the Multistakeholder Impact of Popularity Bias in Recommendation Through Calibration, 2020. _eprint: 2007.12230.

[5] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. The Connection Between Popularity Bias, Calibration, and Fairness in Recommendation, 2020. _eprint: 2008.09273.

[6] Gediminas Adomavicius and YoungOk Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):896–911, 2012.

[7] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[8] Sebastiano Antenucci, Simone Boglio, Emanuele Chioso, Ervin Dervishaj, Shuwen Kang, Tommaso Scarlatti, and Maurizio Ferrari Dacrema. Artist-driven layering and user's behaviour impact on recommendations in a playlist continuation scenario. In *Proceedings of the ACM Recommender Systems Challenge 2018*. ACM, oct 2018.

[9] Mario Arduini, Lorenzo Noci, Federico Pirovano, Ce Zhang, Yash Raj Shrestha, and Bibek Paudel. Adversarial learning for debiasing knowledge graph embeddings. 2020.

[10] Albert-László Barabási and Márton Pósfai. *Network science*. Cambridge University Press, Cambridge, 2016.

[11] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. 2019. `http://www.fairmlbook.org`.

[12] Christine Bauer, Marta Kholodylo, and Christine Strauss. Music Recommender Systems Challenges and Opportunities for Non-Superstar Artists. In *Bled eConference*, 2017.

[13] Nancy Baym, Rachel Bergmann, Raj Bhargava, Fernando Diaz, Tarleton Gillespie, David Hesmondhalgh, Elena Maris, and Christopher Persaud. Making Sense of Metrics in the Music Industries. *International Journal of Communication*, 15(0), 2021.

[14] Lex Beattie, Dan Taber, and Henriette Cramer. Challenges in translating research to practice for evaluating fairness and bias in recommendation systems. In *Proceedings of the 16th ACM Conference*

on Recommender Systems, RecSys '22, page 528–530, New York, NY, USA, 2022. Association for Computing Machinery.

[15] Riju Bhattacharya, Naresh Nagwani, and Sarsij Tripathi. A comparative study of graph kernels and clustering algorithms. *International Journal of Multimedia Data Engineering and Management*, 12:33–48, 01 2021.

[16] Théo Bontempelli, Benjamin Chapus, François Rigaud, Mathieu Morlon, Marin Lorant, and Guillaume Salha-Galvan. Flow moods: Recommending music by moods on deezer. In *RecSys '22*, 2022.

[17] Ludovico Boratto, Gianni Fenu, and Mirko Marras. Interplay between upsampling and regularization for provider fairness in recommender systems. *User Modeling and User-Adapted Interaction*, 2021.

[18] Georgina Born, Jeremy Morris, Fernando Diaz, and Ashton Anderson. Artificial intelligence, music recommendation, and the curation of culture. *Schwartz Reisman Institute for Technology and Society White Paper*, 2021.

[19] Avishek Bose and William Hamilton. Compositional fairness constraints for graph embeddings. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 715–724. PMLR, Jun 2019.

[20] Christopher Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11, 01 2010.

[21] Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[22] Robin Burke and Maryam Ramezani. *Matching Recommendation Technologies and Domains*, pages 367–386. 01 2011.

[23] Rocío Cañamares and Pablo Castells. Should i follow the crowd? a probabilistic analysis of the effectiveness of popularity in recommender systems. In *The 41st International ACM SIGIR Conference on Research ; Development in Information Retrieval*, SIGIR '18, page 415–424, New York, NY, USA, 2018. Association for Computing Machinery.

[24] Zhipeng Cai, Christian Esposito, Tooska Dargahi, and Chaokun Wang. Graph-powered learning for social networks. *Neurocomputing*, 501:244–245, 2022.

[25] Semih Cantürk. Taxonomy of datasets in graph learning : A data-driven approach to improve gnn benchmarking, 2022.

[26] Pablo Castells and Alistair Moffat. Offline recommender system evaluation: Challenges and new directions. *AI Magazine*, 43(2):225–238, 2022.

[27] Òscar Celma and Pedro Cano. From hits to niches? or how popular artists can bias music recommendation and discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, NETFLIX '08, New York, NY, USA, 2008. Association for Computing Machinery.

[28] Òscar Celma and Perfecto Herrera. A new approach to evaluating novel recommendations. In *Proceedings of the ACM Conference on Recommender Systems, RecSys '08*, 2008.

[29] Òscar Celma and Pedro Cano. From Hits to Niches? Or How Popular Artists Can Bias Music Recommendation and Discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, NETFLIX '08, New York, NY, USA, 2008. Association for Computing Machinery. event-place: Las Vegas, Nevada.

[30] Abhijnan Chakraborty, Anikó Hannák, Asia J. Biega, and Krishna P. Gummadi. Fair sharing for sharing economy platforms. In *Fairness, Acountability, and Transparency in Recommender Systems*, 2017.

[31] Ching-Wei Chen, Paul Lamere, Markus Schedl, and Hamed Zamani. Recsys challenge 2018: Automatic music playlist continuation. In *RecSys '18*, 2018.

[32] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and Debias in Recommender System: A Survey and Future Directions, 2020.

[33] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and debias in recommender system: A survey and future directions, 2020.

[34] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: Fast learning with graph convolutional networks via importance sampling, 2018.

[35] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[36] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-GCN. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Jul 2019.

[37] Janneth Chicaiza and Priscila Valdiviezo Díaz. A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. *Inf.*, 12:232, 2021.

[38] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. 2018.

[39] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Transfer learning for music classification and regression tasks. March 2017.

[40] Weilin Cong, Morteza Ramezani, and Mehrdad Mahdavi. On the importance of sampling in training gcns: Tighter analysis and variance reduction, 2021.

[41] Peng Cui, Lingfei Wu, Jian Pei, Liang Zhao, and Xiao Wang. Graph representation learning. In Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao, editors, *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 17–26. Springer Singapore, Singapore, 2022.

[42] Sally Jo Cunningham, David Bainbridge, and Dana Mckay. Finding new music: A diary study of everyday encounters with novel songs. In *International Society for Music Information Retrieval Conference*, 2007.

[43] Aminu Da'u and Naomie Salim. Recommendation system based on deep learning methods: a systematic review and new directions. *Artificial Intelligence Review*, 53:2709–2748, 2019.

[44] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[45] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music, 2020.

[46] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. Evaluating stochastic rankings with expected exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. ACM, Oct 2020.

[47] Yujuan Ding, P.Y. Mok, Yunshan Ma, and Yi Bin. Personalized fashion outfit generation with user coordination preference learning. *Information Processing & Management*, 60(5):103434, 2023.

[48] Yushun Dong, Jian Kang, Hanghang Tong, and Jundong Li. Individual Fairness for Graph Neural Networks: A Ranking Based Approach. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery &; Data Mining*, KDD '21, pages 300–310, New York, NY, USA, 2021. Association for Computing Machinery. event-place: Virtual Event, Singapore.

[49] Yushun Dong, Ninghao Liu, Brian Jalaian, and Jundong Li. EDITS: Modeling and mitigating data bias for graph neural networks. In *Proceedings of the ACM Web Conference 2022*. ACM, Apr 2022.

[50] Allen B. Downey. Evidence for long-tailed distributions in the internet. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, IMW '01, page 229–241, New York, NY, USA, 2001. Association for Computing Machinery.

[51] Eric Drott. Why the next song matters: Streaming, recommendation, scarcity. *Twentieth-Century Music*, 15:325–357, 10 2018.

[52] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 214–226, New York, NY, USA, 2012. Association for Computing Machinery.

[53] Michael D. Ekstrand, F. Maxwell Harper, Martijn C. Willemsen, and Joseph A. Konstan. User perception of differences in recommender algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, page 161–168, New York, NY, USA, 2014. Association for Computing Machinery.

[54] Michael D. Ekstrand, Mucun Tian, Ion Madrazo Azpiazu, Jennifer D. Ekstrand, Oghenemaro Anuyah, David McNeill, and Maria Soledad Pera. All the cool kids, how do they fit in?: Popularity and demographic biases in recommender evaluation and effectiveness. In Sorelle A. Friedler and Christo Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 172–186. PMLR, 23–24 Feb 2018.

[55] Andres Ferraro, Dmitry Bogdanov, Jisang Yoon, KwangSeob Kim, and Xavier Serra. Automatic playlist continuation using a hybrid recommender system combining features from text and audio. In *Proceedings of the ACM Recommender Systems Challenge 2018*. ACM, oct 2018.

[56] Andres Ferraro, Xavier Serra, and Christine Bauer. Break the Loop: Gender Imbalance in Music Recommenders. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, CHIIR '21, pages 249–254, New York, NY, USA, 2021. Association for Computing Machinery. event-place: Canberra ACT, Australia.

[57] Jonathan Foote. Content-based retrieval of music and audio. 1997.

[58] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, and Yong Li. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Trans. Recomm. Syst.*, 1(1), mar 2023.

[59] Jean Garcia-Gathright, Brian St. Thomas, Christine Hosey, Zahra Nazari, and Fernando Diaz. Understanding and evaluating user satisfaction with music discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 55–64, New York, NY, USA, 2018. Association for Computing Machinery.

[60] Josh Gardner, Simon Durand, Daniel Stoller, and Rachel Bittner. Llark: A multimodal foundation model for music. *arXiv preprint arXiv:2310.07160*, 2023.

[61] Thomas Gaudelet, Ben Day, Arian R Jamasb, Jyothish Soman, Cristian Regep, Gertrude Liu, Jeremy B R Hayter, Richard Vickers, Charles Roberts, Jian Tang, David Roblin, Tom L Blundell, Michael M Bronstein, and Jake P Taylor-King. Utilizing graph machine learning within drug discovery and development. *Briefings in Bioinformatics*, 22(6):bbab159, May 2021. _eprint: https://academic.oup.com/bib/article-pdf/22/6/bbab159/41087478/bbab159.pdf.

[62] Gijs Geleijnse, Markus Schedl, and Peter Knees. The quest for ground truth in musical artist tagging in the social web era. In *International Society for Music Information Retrieval Conference, ISMIR '07*, 2007.

[63] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 06–11 Aug 2017.

[64] Abba Suganda Girsang, Antoni Wibowo, and Edwin. Song Recommendation System Using Collaborative Filtering Methods. In *Proceedings of the 2019 The 3rd International Conference on Digital Technology in Education*, ICDTE 2019, pages 160–162, New York, NY, USA, 2019. Association for Computing Machinery. event-place: Yamanashi, Japan.

[65] Sharad Goel, Andrei Broder, Evgeniy Gabrilovich, and Bo Pang. Anatomy of the long tail: Ordinary people with extraordinary tastes. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, page 201–210, New York, NY, USA, 2010. Association for Computing Machinery.

[66] Yuan Gong, Yu-An Chung, and James Glass. AST: Audio Spectrogram Transformer. In *Proc. Interspeech 2021*, pages 571–575, 2021.

[67] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 855–864, New York, NY, USA, 2016. Association for Computing Machinery.

[68] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. volume 34, pages 3549–3568, 2022.

[69] William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159.

[70] William L. Hamilton. Graph representation learning. volume 14, pages 1–159. Morgan and Claypool, 2020.

[71] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[72] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), dec 2015.

[73] Chuan He, Cong Wang, Yi-Xin Zhong, and Rui-Fan Li. A survey on learning to rank. In *2008 International Conference on Machine Learning and Cybernetics*, 2008.

[74] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR '16*, 2016.

[75] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd*

International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20, page 639–648, New York, NY, USA, 2020. Association for Computing Machinery.

[76] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, jan 2004.

[77] Hesmondhalgh, Raquel Campos Valverde, D. Bondy Valdovinos Kaye, and Zhongwei Li. The impact of algorithmically driven recommendation systems on music consumption and production - a literature review. Technical report, Department for Culture, Media & Sport, United Kingdom, February 2023.

[78] Imran Hossain, Md Aminul Haque Palash, Anika Tabassum Sejuty, Noor A Tanjim, MD Abdullah AL Nasim, Sarwar Saif, Abu Bokor Suraj, Md Mahim Anjum Haque, and Nazmul Karim. A survey of recommender system techniques and the ecommerce domain, 2023.

[79] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. pages 263–272, 12 2008.

[80] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[81] Tim Ingham. Over 60,000 tracks are now uploaded to spotify every day, 2021.

[82] Amir H. Jadidinejad, Craig Macdonald, and Iadh Ounis. How sensitive is recommendation systems' offline evaluation to popularity? In *In Workshop on Offline Evaluation for Recommender Systems (REVEAL2019) at the 13th ACM Conference on Recommender Systems.*, 2019.

[83] Amir Hossein Jadidinejad, Craig Macdonald, and Iadh Ounis. How sensitive is recommendation systems' offline evaluation to popularity? 2019.

[84] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. What Recommenders Recommend: An Analysis of Recommendation Biases and Possible Countermeasures. *User Modeling and User-Adapted Interaction*, 25(5):427–491, December 2015. Place: USA Publisher: Kluwer Academic Publishers.

[85] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, Oct 2002.

[86] Shuyi Ji, Yifan Feng, Rongrong Ji, Xibin Zhao, Wanwan Tang, and Yue Gao. Dual channel hypergraph collaborative filtering. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 2020–2029, New York, NY, USA, 2020. Association for Computing Machinery.

[87] Iman Kamehkhosh and Dietmar Jannach. User perception of next-track music recommendations. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, UMAP '17, page 113–121, New York, NY, USA, 2017. Association for Computing Machinery.

[88] Jian Kang, Jingrui He, Ross Maciejewski, and Hanghang Tong. Inform: Individual fairness on graph mining. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 379–389, New York, NY, USA, 2020. Association for Computing Machinery.

[89] Jian Kang, Yan Zhu, Yinglong Xia, Jiebo Luo, and Hanghang Tong. Rawlsgcn: Towards rawlsian difference principle on graph convolutional network. In *WWW '22*, 2022.

[90] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 321–328. AAAI Press, 2003.

[91] Harsh Khatter, Nishtha Goel, Naina Gupta, and Muskan Gulati. Movie recommendation system using cosine similarity with sentiment analysis. In *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*, pages 597–603, 2021.

[92] Jaehun Kim, Julián Urbano, Cynthia C. S. Liem, and Alan Hanjalic. One deep music representation to rule them all? A comparative analysis of different representation learning strategies. *Neural Computing and Applications*, 32(4):1067–1093, February 2020.

[93] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[94] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores, 2016.

[95] Filip Korzeniowsky, Sergio Oramas, and Fabien Gouyon. Artist similarity with graph neural networks. In *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, 2021.

[96] Matevž Kunaver and Tomaž Požrl. Diversity in recommender systems – A survey. *Knowledge-Based Systems*, 123:154–162, 2017.

[97] Matt J. Kusner, Joshua R. Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness, 2017.

[98] Preethi Lahoti, Krishna P. Gummadi, and Gerhard Weikum. Operationalizing individual fairness with pairwise fair representations. *Proceedings of the VLDB Endowment*, 13(4):506–518, Dec 2019.

[99] Charilaos Lavranos, Petros Kostagiolas, and Konstantina Martzoukou. *Theoretical and Applied Issues on the Impact of Information on Musical Creativity: An Information Seeking Behavior Perspective*, pages 1 – 16. 06 2016.

[100] Jure Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. *Mining of Massive Datasets*. Cambridge University Press, 2020.

[101] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32, 01 2018.

[102] Yunqi Li, Hanxiong Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. User-oriented fairness in recommendation. In *Proceedings of the Web Conference 2021*. ACM, Apr 2021.

[103] Yunqi Li, Hanxiong Chen, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. Towards personalized fairness based on causal notion. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Jul 2021.

[104] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[105] Xin Liu, Mingyu Yan, Lei Deng, Guoqi Li, Xiaochun Ye, and Dongrui Fan. Sampling methods for efficient training of graph convolutional networks: A survey, 2021.

[106] Kachun Lo and Tsukasa Ishigaki. Matching novelty while training: Novel recommendation based on personalized pairwise loss weighting. *2019 IEEE International Conference on Data Mining (ICDM)*, pages 468–477, 2019.

[107] Jing Ma, Ruocheng Guo, Mengting Wan, Longqi Yang, Aidong Zhang, and Jundong Li. Learning fair node representations with graph counterfactual fairness. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. ACM, Feb 2022.

[108] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. A graph-based approach for mitigating multi-sided exposure bias in recommender systems. *ACM Transactions on Information Systems*, 40(2):1–31, nov 2021.

[109] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness &; satisfaction in recommendation systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 2243–2251, New York, NY, USA, 2018. Association for Computing Machinery.

[110] Alessandro Melchiorre, Navid Rekabsaz, Emilia Parada-Cabaleiro, Stefan Brandl, Oleg Lesota, and Markus Schedl. Investigating gender fairness of recommendation algorithms in the music domain. *Information Processing & Management*, 2021.

[111] Lien Michiels, Jens Leysen, Annelien Smets, and Bart Goethals. What are filter bubbles really? a review of the conceptual and empirical work. In *Adjunct Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization*, UMAP '22 Adjunct, page 274–279, New York, NY, USA, 2022. Association for Computing Machinery.

[112] Judith Moeller, Damian Trilling, Natali Helberger, and Bram Es. Do not blame it on the algorithm: an empirical assessment of multiple recommender systems and their impact on content diversity. *Information, Communication Society*, 21:1–19, 03 2018.

[113] Lillio Mok, Samuel F. Way, Lucas Maystre, and Ashton Anderson. The dynamics of exploration on spotify. *Proceedings of the International AAAI Conference on Web and Social Media*, 16(1):663–674, May 2022.

[114] Matti Mäntymäki and Najmul Islam. Gratifications from using freemium music streaming services: Differences between basic and premium users. In *International Confererence on Information Systems*, 12 2015.

[115] M. E. J. Newman. *Networks: an introduction.* Oxford University Press, Oxford; New York, 2010.

[116] Taehyung Noh, Haein Yeo, Myungjin Kim, and Kyungsik Han. A study on user perception and experience differences in recommendation results by domain expertise: The case of fashion domains. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI EA '23, New York, NY, USA, 2023. Association for Computing Machinery.

[117] Marcus O'Dair and Andrew Fry. *Beyond the black box in music streaming: the impact of recommendation systems upon artists*, pages 35–47. 09 2020.

[118] Sergio Oramas, Oriol Nieto, Francesco Barbieri, and Xavier Serra. Multi-label music genre classification from audio, text, and images using deep features. 07 2017.

[119] Sergio Oramas, Mohamed Sordo, Luis Espinosa-Anke, and Xavier Serra. A Semantic-based Approach for Artist Similarity. October 2015.

[120] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[121] Jiyoung Park, Jongpil Lee, Jangyeon Park, Jung-Woo Ha, and Juhan Nam. Representation Learning of Music Using Artist Labels, 2017.

[122] Yoon-Joo Park and Alexander Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, page 11–18, New York, NY, USA, 2008. Association for Computing Machinery.

[123] Gourab K Patro, Arpita Biswas, Niloy Ganguly, Krishna P. Gummadi, and Abhijnan Chakraborty. FairRec: Two-sided fairness for personalized recommendations in two-sided platforms. In *Proceedings of The Web Conference 2020*. ACM, apr 2020.

[124] Mathieu Prang. *Representation learning for symbolic music*. Theses, Sorbonne Université, June 2021. Issue: 2021SORUS489.

[125] Robert Prey. Locating power in platformization: Music streaming playlists and curatorial power. *Social Media + Society*, 6:205630512093329, 04 2020.

[126] Michael Pulis and Josef Bajada. Siamese Neural Networks for Content-Based Cold-Start Music Recommendation. In *Fifteenth ACM Conference on Recommender Systems*, pages 719–723. Association for Computing Machinery, New York, NY, USA, 2021.

[127] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. Sequence-aware recommender systems. volume 51, New York, NY, USA, Jul 2018. Association for Computing Machinery.

[128] Andreas Raff, Andreas Mladenow, and Christine Strauss. Music discovery as differentiation strategy for streaming providers. In *Proceedings of the 22nd International Conference on Information Integration and Web-Based Applications & Services*, iiWAS '20, page 476–480, New York, NY, USA, 2021. Association for Computing Machinery.

[129] Heritiana Renaud Ranaivoson. *Online platforms and cultural diversity in the audiovisual sectors: a combined look at concentration and algorithms*, pages 100–118. Routledge Studies in Media and Cultural Industries. Routledge International, 2019.

[130] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2012.

[131] Denise Rey and Markus Neuhäuser. Wilcoxon-signed-rank test. 2011.

[132] Wondo Rhee, Sung Min Cho, and Bongwon Suh. Countering popularity bias by regularizing score differences. In *Proceedings of the 16th ACM Conference on Recommender Systems*, RecSys '22, page 145–155, New York, NY, USA, 2022. Association for Computing Machinery.

[133] Guillaume Salha-Galvan. *Contributions to Representation Learning with Graph Autoencoders and Applocations to Music Recommendation*. Theses, Institut Polytechnique de Paris, March 2022.

[134] Guillaume Salha-Galvan, Romain Hennequin, Benjamin Chapus, Viet-Anh Tran, and Michalis Vazirgiannis. Cold Start Similar Artists Ranking with Gravity-Inspired Graph Autoencoders, 2021.

[135] Guillaume Salha-Galvan, Romain Hennequin, Benjamin Chapus, Viet-Anh Tran, and Michalis Vazirgiannis. Cold start similar artists ranking with gravity-inspired graph autoencoders, 2021.

[136] Antonia Saravanou, Federico Tomasi, Rishabh Mehrotra, and Mounia Lalmas. Multi-Task Learning of Graph-based Inductive Representations of Music Content. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 602–609, Online, November 2021. ISMIR.

[137] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval*, 7(2):95–116, apr 2018.

[138] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation, 2016.

[139] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77):2539–2561, 2011.

[140] Sigurdur Sigurdsson, Kaare Petersen, and Tue Lehn-Schiøler. Mel frequency cepstral coefficients: An evaluation of robustness of mp3 encoded music. In *7th International Society for Music Information Retrieval Conference (ISMIR 2006),*, pages 286–289, 01 2006.

[141] Daniel A. Spielman. Spectral graph theory and its applications. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, 2007.

[142] Janne Spijkervet and John Ashley Burgoyne. Contrastive learning of musical representations, 2021.

[143] Darko Stanisljevic. The impact of spotify features on music discovery in the streaming platform age. Master's thesis, July 2020.

[144] Harald Steck. Item popularity and recommendation accuracy. In *RecSys '11*. ACM, 2011.

[145] Ja-Hwung Su, Hsin-Ho Yeh, Philip Yu, and Vincent Tseng. Music recommendation using content and context information mining. *Intelligent Systems, IEEE*, 25:16 – 26, 03 2010.

[146] Lei Tang and Huan Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, page 817–826, New York, NY, USA, 2009. Association for Computing Machinery.

[147] Lei Tang and Huan Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, page 817–826, New York, NY, USA, 2009. Association for Computing Machinery.

[148] Rianne van den Berg, Thomas N. Kipf, and Max Welling. Graph convolutional matrix completion. In *Proceedings of the 24nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18. ISMIR, 2018.

[149] Petar Veličković.

[150] Quentin Villermet, Jérémie Poiroux, Manuel Moussallam, Thomas Louail, and Camille Roth. Follow the guides: disentangling human and algorithmic curation in online music consumption. In *RecSys '21*. ACM, 2021.

[151] S.V.N. Vishwanathan, Nicol N. Schraudolph, Risi Kondor, and Karsten M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11(40):1201–1242, 2010.

[152] Maksims Volkovs, Himanshu Rai, Zhaoyue Cheng, Ga Wu, Yichao Lu, and Scott Sanner. Two-stage model for automatic playlist continuation at scale. In *Proceedings of the ACM Recommender Systems Challenge 2018*, RecSys Challenge '18, New York, NY, USA, 2018. Association for Computing Machinery.

[153] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks, 2019.

[154] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z. Sheng, Mehmet A. Orgun, Longbing Cao, Francesco Ricci, and Philip S. Yu. Graph Learning based Recommender Systems: A Review, 2021. _eprint: 2105.06339.

[155] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, jul 2019.

[156] Xiuling Wang and Wendy Hui Wang. Providing item-side individual fairness for deep recommender systems. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 117–127, New York, NY, USA, 2022. Association for Computing Machinery.

[157] Zhaokang Wang, Yunpan Wang, Chunfeng Yuan, Rong Gu, and Yihua Huang. Empirical analysis of performance bottlenecks in graph neural network training and inference with gpus. *Neurocomputing*, 446:165–191, 2021.

[158] Shiqi Wei and Gus Xia. Learning long-term music representations via hierarchical contextual constraints, 2022.

[159] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 1791–1800, New York, NY, USA, 2021. Association for Computing Machinery.

[160] Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. Personalized news recommendation: Methods and challenges, 2022.

[161] Haolun Wu, Chen Ma, Bhaskar Mitra, Fernando Diaz, and Xue Liu. Multi-fr: A multi-objective optimization method for achieving two-sided fairness in e-commerce recommendation. *ArXiv*, abs/2105.02951, 2021.

[162] Jiancan Wu, Xiangnan He, Xiang Wang, Qifan Wang, Weijian Chen, Jianxun Lian, and Xing Xie. Graph convolution machine for context-aware recommender system. *Frontiers of Computer Science*, 16(6), jan 2022.

[163] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: A survey, 2020.

[164] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: A survey, 2020.

[165] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: A survey. *ACM Comput. Surv.*, 55(5), dec 2022.

[166] Yusong Wu*, Ke Chen*, Tianyu Zhang*, Yuchen Hui*, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2023.

[167] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, Jan 2021.

[168] Christopher C. Yang, Hsinchun Chen, and Kay Hong. Visualization of large category map for internet browsing. *Decis. Support Syst.*, 35(1):89–102, apr 2003.

[169] Liangwei Yang, Shengjie Wang, Yunzhe Tao, Jiankai Sun, Xiaolong Liu, Philip S. Yu, and Taiqing Wang. DGRec: Graph neural network for recommendation with diversified embedding generation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. ACM, feb 2023.

[170] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Jul 2018.

[171] Hamed Zamani, Markus Schedl, Paul Lamere, and Ching-Wei Chen. An analysis of approaches taken in the acm recsys challenge 2018 for automatic music playlist continuation, 2019.

[172] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. FA∗IR. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM '17'*. ACM, Nov 2017.

[173] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Accurate, efficient and scalable graph embedding. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2019.

[174] An Zhang, Wenchang Ma, Xiang Wang, and Tat seng Chua. Incorporating bias-aware margins into contrastive loss for collaborative filtering. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2022.

[175] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system. *ACM Computing Surveys*, 52(1):1–38, Jan 2020.

[176] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. Causal intervention for leveraging popularity bias in recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 11–20, New York, NY, USA, 2021. Association for Computing Machinery.

[177] Yixiao Zhang, Ziyu Wang, Dingsu Wang, and Gus Xia. Butter: A representation learning framework for bi-directional music-sentence retrieval and generation. In *NLP4MUSA*, November 2020.

[178] Minghao Zhao, Le Wu, Yile Liang, Lei Chen, Jian Zhang, Qilin Deng, Kai Wang, Xudong Shen, Tangjie Lv, and Runze Wu. Investigating accuracy-novelty performance for graph-based collaborative filtering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, jul 2022.

[179] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*, WWW '21, page 2980–2991, New York, NY, USA, 2021. Association for Computing Machinery.

[180] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. Popularity-opportunity bias in collaborative filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 85–93, New York, NY, USA, 2021. Association for Computing Machinery.

[181] Chenyi Zhuang and Qiang Ma. Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 499–508, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

[182] Difan Zou, Ziniu Hu, Yewen Wang, Song Jiang, Yizhou Sun, and Quanquan Gu. Layer-dependent importance sampling for training deep and large graph convolutional networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2019.