Towards Trustworthy Digital Ecosystem: From Fair Representation Learning to Fraud Detection

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Arvindh A
2019111010
arvindh.a@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
March 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled **"Towards Trustworthy Digital Ecosystem: From Fair Representation Learning to Fraud Detection"** by Arvindh A, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Ponnurangam Kumaraguru

To my parents and friends

Acknowledgments

I want to begin by expressing my deepest gratitude to my advisor, Prof. PK, for introducing me to the world of research and empowering me with opportunities to explore, learn, and take ownership of my work. His guidance has been invaluable in shaping not only my academic skills but also for my personal growth. I also admire his commitment to not only academically helping the students but also going beyond it and becoming a life mentor.

I am also sincerely grateful to the remarkable collaborators I've had the privilege of working with over the years – Dr. Ramasuri Narayanam, Prof. Jisun An, and Prof. Polo Chau, among others. These interactions broadened my horizons and honed my technical skills. Informal conversations with them helped refine my research interests and better understand the intricacies of the academic world.

My heartfelt thanks go to my family for their unwavering support. I am truly fortunate to have parents who encourage my academic pursuits and motivate me even during my most challenging times. Their belief in my decisions, gentle guidance towards independence, and ability to step in when needed are pivotal reasons for my growth.

I'm very grateful to my late friend, Arjun, for being one of the strongest motivators for me to indulge in this area of research, guiding me during the advisor search, and being an amazing friend in general. Although I don't express it very often, you are most definitely missed and will always be an integral part of my academic journey. I also thank Rengan for his immense support and guidance in helping me choose the right path. I'm indebted to the Precog group for their constant support and the incredible friendships I've formed. The platform it provided to meet and exchange ideas with like-minded people served as my primary source of motivation and a pillar of support. My gratitude extends especially to Aakash Aanegola and Amul Agrawal, for being great friends and exceptional co-authors. Furthermore, I would like to thank IIIT for providing an exceptional platform for my academic and research pursuits. Finally, thank you to all my friends who made this journey so enjoyable, I will definitely miss the pizza nights and our late-night dives in random internet rabbit holes.

Abstract

Two critical challenges arise in the interconnected realm of online platforms: the need to ensure equitable representation for entities and the emphasis on identifying deceptive practices. This work aims to address both by introducing CAFIN to decrease the disparity in the representations of GNNs, and by detecting and studying anomalies on the Google Play Store reviewer network for fraud prevention, moving in the direction of reshaping digital ecosystems to be both fair and trustworthy.

Unsupervised Representation Learning on graphs is gaining traction due to the increasing abundance of unlabelled network data and the compactness, richness, and usefulness of the representations generated. In this context, the need to consider fairness and bias constraints while generating the representations has been well-motivated and studied to some extent in prior works. One major limitation of most of the prior works in this setting is that they do not aim to address the bias generated due to connectivity patterns in the graphs, such as varied node centrality, which leads to a disproportionate performance across nodes. In our work, we aim to address this issue of mitigating bias due to inherent graph structure in an unsupervised setting. To this end, we propose CAFIN, a centrality-aware fairness-inducing framework that leverages the structural information of graphs to tune the representations generated by existing frameworks. We deploy it on GraphSAGE (a popular framework in this domain) and showcase its efficacy on two downstream tasks - Node Classification and Link Prediction. Empirically, CAFIN consistently reduces the performance disparity across popular datasets (varying from 18% to 80% reduction in performance disparity) from various domains while incurring only a minimal cost of fairness.

Google Play Store's policy forbids the use of incentivized installs, ratings, and reviews to manipulate the placement of apps. However, there still exist apps that incentivize installs for other apps on the platform. To understand how install-incentivizing apps affect users, we examine their ecosystem through a socio-technical lens and perform a mixed-methods analysis of their reviews and permissions. Our dataset contains 319K reviews collected daily over five months from 60 such apps that cumulatively account for over 160.5M installs. We perform qualitative analysis of the reviews to reveal various types of dark patterns that developers incorporate in install-incentivizing apps, highlighting their normative concerns at both user and platform levels. Permissions requested by these apps validate our discovery of dark patterns, with over 92% apps accessing sensitive user information. We find evidence of fraudulent reviews on install-incentivizing apps, following which we model them as an edge stream in a dynamic bipartite graph of apps and reviewers. Our proposed reconfiguration of a state-of-the-art microcluster anomaly detection algorithm yields promising preliminary results in detecting this fraud. We discover

highly significant lockstep behaviors exhibited by reviews that aim to boost the overall rating of an install-incentivizing app. Upon evaluating the 50 most suspicious clusters of boosting reviews detected by the algorithm, we find (i) near-identical pairs of reviews across 94% (47 clusters), and (ii) over 35% (1,687 of 4,717 reviews) present in the same form near-identical pairs within their cluster. We also discuss how fraud is intertwined with labor and poses a threat to the trust and transparency of Google Play.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent technological advancements have transformed how we interact, share information, and make decisions. Yet, the increasing complexity and lack of transparency in many technological systems raise concerns about fairness, accountability, and the preservation of user autonomy. While these advancements promise societal progress, they also risk amplifying existing inequalities or setting up the stage for new forms of subtle manipulation and fraud. This thesis investigates two interconnected aspects of safety within the digital landscape: the tendency of algorithmic systems to produce biased outputs favoring one party over the others [O'Neil, 2016], and the exploitation of users through deceptive design practices [Mathur et al., 2019] and the identification of groups of fraudulent users acting in synchronization to disrupt the platform.

With the rising adoption of deep models, the challenge of tackling algorithmic bias is paramount. In the supervised setting, models trained on data that reflect real-world disparities may perpetuate and amplify those biases [Hardt et al., 2016a]. Even unsupervised models tend to get misled by the stereotypical patterns [Buet-Golfouse and Utyagulov, 2022] observed in the data and the architecture's pitfalls themselves. This can lead to discriminatory outcomes in high-stakes domains, such as loan approvals or predictive policing, with far-reaching societal consequences. The first work in this thesis aims to address this issue by developing a framework, CAFIN (Centrality Aware Fairness Inducing IN-processing for Unsupervised Representation Learning on Graphs), designed to promote fairness and mitigate bias within a large set of graph representation learning methods. CAFIN aims to mitigate the bias in the final embeddings induced due to the centrality of different data points within the overall graph structure by modifying the loss functions. By ensuring fair representation learning, we hope to lay the groundwork for downstream machine learning tasks that are less likely to exhibit discriminatory behavior.

Simultaneously, the widespread use of "dark patterns" [Mildner et al., 2023] and the proliferation of fraud-induced content in user-facing platforms pose a significant ethical challenge. These deceptive design strategies manipulate users into behaviors that may contradict their interests, subtly eroding trust and exploiting the users' vulnerabilities. The second work in this thesis investigates this phenomenon, analyzing in detail how specific design strategies influence user actions, and how to identify micro-clusters of users who do targeted review boosts and attacks on apps.

Ultimately, this thesis aims to tie two important problems to foster a deeper understanding of the ethical complexities inherent in the digital landscape. The hope is to advance the ongoing efforts to promote transparency, fairness, and user autonomy in an era of increasingly sophisticated and complex online interaction platforms. By addressing the challenges of algorithmic bias, deceptive design practices, and the identification of fake review networks, the hope is to work towards a future where technology empowers individuals and fosters a more just and equitable society. Chapters 2, 3, 4 cover CAFIN, Chapter 5 covers our work on exposing dark patterns and identifying microclusters of anomalous users on Google Play Store, and finally, Chapter 6 covers the conclusion, limitations, and the scope for future work.

Chapter 2

The lack and therefore the need for Fair Representations [1]

Due to the prevalence and popularity of online social networks, network data has grown significantly, both in quantity and quality, over the years [Leskovec et al., 2020]. Such rich data can be exploited to gather information at both the individual and community levels. The influx of data having inter-personal connections (represented as graphs) has served as motivation to develop several unsupervised learning algorithms for various tasks on graphs [Hamilton et al., 2017, Velickovic et al., 2018]. These methods leverage node features along with neighborhood information to learn node representations that do not depend on the domain of the underlying graph or the desired task at hand.

It is essential that these node representations are generated with appropriate fairness measures, especially in the context of real-world deployments, to minimize bias induced by these graph learning frameworks on downstream tasks. Accordingly, *fairness* in the context of trained decision-making systems has increased in popularity recently due to the numerous social distresses caused when systems not incorporating adequate fairness measures were deployed in the wild [Pessach and Shmueli, 2022, Mehrabi et al., 2021]. The job platform XING is an extreme example that exhibited gender-based discrimination [Choudhary et al., 2022].

Previous works aimed to mitigate such unfairness, in this context, focus on ensuring minimal disparity in performance among individuals or groups defined by some membership criteria. Although sensitive node attributes generally decide these group memberships, a recent uptick in research considers intrinsic node properties, specifically node degree, to evaluate the fairness of Graph Neural Networks (GNNs). For example, recent work [Tang et al., 2020] provides theoretical proof that a popular subclass of graph neural networks − graph convolutional networks (GCNs) − are biased (in performance) towards high-degree nodes. They propose a degree-specific GCN layer targetting degree unfairness in the model and design a self-supervised learning algorithm for attaching pseudo labels to unlabelled nodes, which further helps low-degree nodes to perform better. Later, RawlsGCN [Kang et al., 2022] reveals

[1] The next 3 chapters are a part of *Arvindh Arun, Aakash Aanegola, Amul Agrawal, Ramasuri Narayanam, and Ponnurangam Kumaraguru.* ***CAFIN: Centrality Aware Fairness Inducing IN-processing for unsupervised representation learning on graphs.*** *In ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Krakow, Poland, volume 372 of Frontiers in Artificial Intelligence and Applications, pages 101 - 108. IOS Press, 2023.*

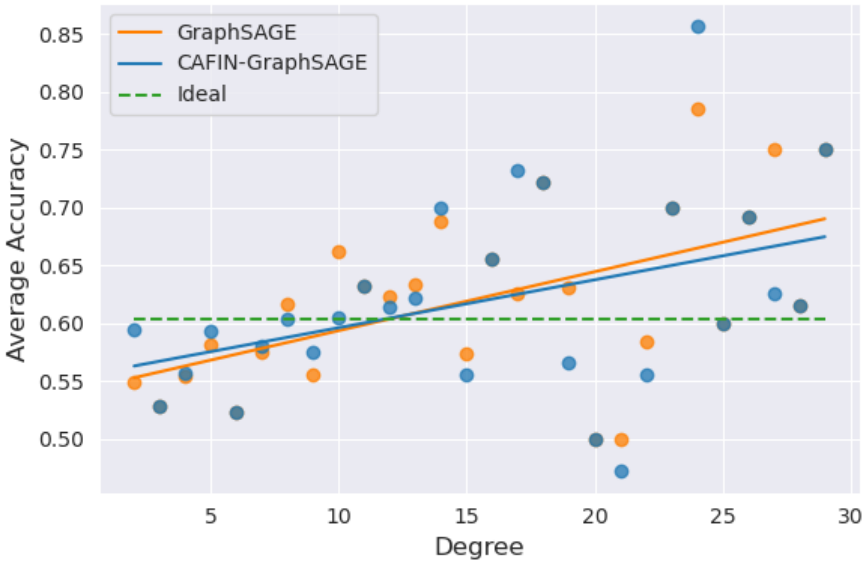

Figure 2.1 Degree vs. Accuracy plot (Twitch dataset) - For the original GraphSAGE on the Node Classification task, the accuracy increases steadily with the degree (slope=0.0051). After the introduction of CAFIN, the slope decreases significantly (slope=0.0041, a 20% reduction), leading to lower performance disparity between high and low-degree nodes with negligible reduction (-0.3%) in the overall accuracy.

the root cause of this degree-related unfairness by analyzing the gradients of weight matrices in GCN and proposes techniques to mitigate this bias.

GNNs refine node embeddings by aggregating information from their neighbors. So, the efficacy of a node's representation is bound to be correlated to its abundance of structural information [Liu et al., 2021]. This correlation creates a disparity in the richness of embeddings between structurally rich nodes (highly central) and the rest (less central). Figure 2.1 empirically corroborates this claim. This disparity is even more concerning as the centralities (degree) of most real-world graphs follow the power-law distribution. This implies that a major fraction of nodes have low centrality scores and hence deficient representations compared to a small fraction of nodes having high centrality.

Most of the works in the literature focus on imposing fairness concerning sensitive attributes but often overlook the more inherent centrality-induced disparity. Recent works [Liu et al., 2023] also probe into how masking the sensitive attributes may not be enough, as some of the characteristics can seep into the inherent network structure. Our work in this paper focuses exclusively on reducing the performance disparity induced among groups of nodes due to skewed centrality distributions. Towards this end, we propose a generalized (additive) modification to the loss function of well-known unsupervised GNNs to impose group fairness constraints while minimizing the cost induced by the same. To formally demonstrate our approach, we consider GraphSAGE [Hamilton et al., 2017] – a popular unsupervised graph learning framework and widely adopted in many domains [Ying et al., 2018, Lo et al., 2022, Liu

et al., 2020] − and then show how we extend its objective function with fairness constraints. GraphSAGE, as studied empirically, focuses more on less frequent higher-degree nodes than on more frequent lower-degree nodes, leading to a performance disparity between the two groups of nodes. We remedy this limitation of GraphSAGE through our work.

Note that these fairness constraints can be added to any underlying graph learning algorithm at three different stages: before learning (Pre-processing), during learning (In-processing), and after learning (Post-processing) [Mehrabi et al., 2021]. In-processing is considered robust and generalizable and finds its application across various domains as it directly adds a secondary objective to the original [Zafar et al., 2019]; hence we adopt this technique in our proposed framework.

In particular, we propose a framework, Centrality Aware Fairness inducing IN-processing (CAFIN), that focuses on augmenting the unsupervised version of GraphSAGE to induce centrality-based (ex: degree) group fairness as an objective while maintaining similar performance on downstream tasks. *To the best of our knowledge, CAFIN is the first work to deal with centrality-driven fairness for unsupervised graph learning, as all other methods work in the supervised or semi-supervised setting (and also largely do not tackle centrality-based fairness aspects).* Thus, our primary contribution is *a novel in-processing technique to achieve centrality-aware group fairness for unsupervised graph node representation learning.*

The following section reviews relevant literature on (unsupervised) graph representation learning and existing fairness measures for these graph representation learning algorithms.

Graph Representation Learning. Unsupervised representation learning on graphs has seen a recent explosion due to the availability of unlabelled structured graph data [Velickovic et al., 2018, Hamilton et al., 2017]. In specific, GraphSAGE [Hamilton et al., 2017], a method that samples and aggregates information from node neighbors has found extensive applications in recommender systems [Ying et al., 2018], intrusion detection systems [Lo et al., 2022], traffic networks [Liu et al., 2020], and more due to its versatility and applicability on large graphs.

GraphSAGE [Hamilton et al., 2017] is a popular inductive representation learning framework specifically tailored for efficient performance on large networks. Instead of training feature representations for each node in the graph, it learns a set of functions that aggregate feature information from the neighborhood of a node to update the node representation, helping it learn node feature embeddings while accounting for information flow from neighbors. It also uses a constrastive-learning based unsupervised loss function for learning embeddings in a task-agnostic fashion, which removes the dependence of network parameters on downstream tasks. It functions efficiently because of the random sampling in each stage of the pipeline, drastically reducing the training time as only a subset of the node neighborhood is utilized. The downside of random sampling is that it induces stochasticity in the learned embeddings, making them highly volatile and dependent on the random seed used during training [Schumacher et al., 2020].

Fairness in Graph Learning Algorithms. The influx of deep learning technologies into the real-world setting and them leading to possibly undesired conclusions has prompted the inquisition into the

fairness of the algorithms. More specific to graphs, studies like [Bertrand and Mullainathan, 2004, Oreopoulos, 2011] explore the fairness of algorithms used for recruitment, and similarly [Khajehnejad et al., 2020] explore the issue of the unfair impact of influential nodes on the overall graph and introduce performance disparity.

The disparity introduced by these algorithms is quantifiable, and there are two primary methods to evaluate the fairness of a graph learning algorithm - individual and group fairness. Individual fairness seeks to attain similar treatment for similar individuals [Dwork et al., 2012], whereas group fairness aims to reduce the bias that algorithms tend to possess towards certain groups [Hardt et al., 2016b]. Group membership is usually defined based on sensitive node attributes like gender, race, and economic background in most studies [Krasanakis et al., 2021]. However, since a lot of graph data is unlabelled or does not possess sensitive node attributes, this information may not always be available. In contrast, very few studies like [Avin et al., 2015] divide them based on node characteristics like centrality - characteristics intrinsic to the graph and present irrespective of domain. Furthermore, [Karimi et al., 2018] confirms that degree disparities exist in real social networks, encouraging us to alleviate disparities based on intrinsic node attributes to achieve fairness.

Previous work seeks to make graph algorithms fair by (a) preprocessing the original graph to remove potential bias, for example, FairDrop [Spinelli et al., 2021] that adds and removes edges to induce fairness, thereby altering graph structure, (b) in-processing during the training phase, for example, [Kang et al., 2022] that modifies the gradient used in the optimization, and (c) postprocessing the node embeddings to remove bias [Mehrabi et al., 2021].

Several metrics have been proposed to evaluate group fairness along with the proliferation of methods to augment fairness. [Newman, 2003] present the Assortative Mixing Coefficient, which measures communities' dependence on protected attributes and models relations between communities where connections are considered fair when the coefficient is 0. The notion of average statistical imparity [Kang et al., 2021] computes the performance differences across groups but primarily caters to the two-group setting. We extend the notion of imparity to different tasks through minor modifications and utilize it for evaluation.

Previous works utilize in-processing techniques for fairer results, like [Kang et al., 2022] that uses the Rawlsian difference principle to mitigate unfairness across the degree of Graph Convolutional Networks (GCN) and [Liu et al., 2021], which learns robust tail node (low-degree) representations by transferring information from central nodes. We address similar concerns in an unsupervised setting through CAFIN, as most prior works focus primarily on supervised and semi-supervised variants.

Chapter 3

CAFIN

This chapter covers important concepts contextualizing our work and proposes our new centrality-driven fairness framework for unsupervised graph representation learning.

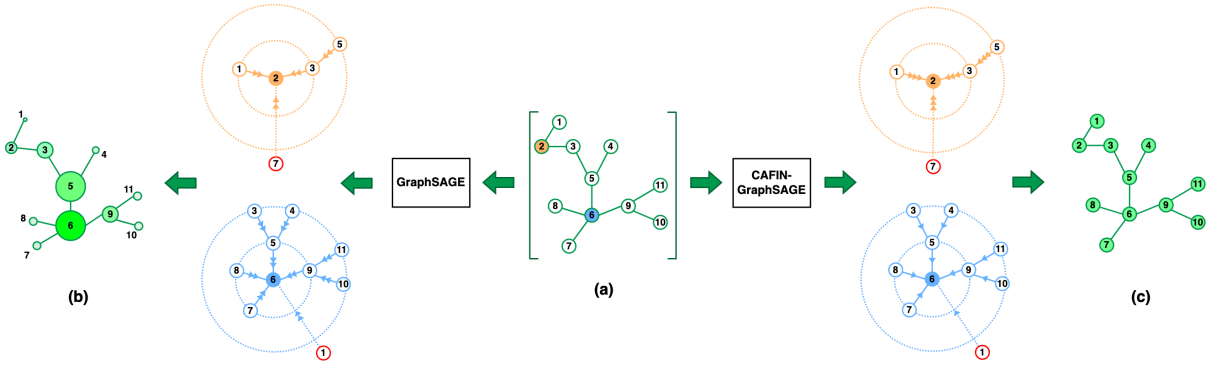

Figure 3.1 Visual Depiction of CAFIN - GraphSAGE refines node embeddings using positive and negative samples as mentioned in section 3.1.2. In the input graph (a), node 6 is a high-degree node (popular), while node 2 has a low-degree (unpopular). As it can be noted from their computation graphs (blue and orange), node 6 has richer structural information when compared to node 2, which causes a disparity in the information flow (indicated by the arrow directions). This transitively causes a disparity in the final quality of the representations learned. The node sizes in graphs (b) and (c) represent the performance distribution in downstream tasks using the final representations learned. The introduction of CAFIN prioritizes the information flow in the computation graphs of less central nodes (indicated by stronger arrows) by penalizing them more, leading to a more homogenous distribution performance in downstream tasks, as shown in graph (c).

3.1 Preliminaries

We provide a brief overview of unsupervised representation learning, focusing on GraphSAGE, followed by group fairness (which we work with in this paper) and the evaluation metric for fairness.

3.1.1 Unsupervised Representation Learning

Unsupervised Representation Learning involves learning useful and rich features from unlabeled data. The learned representations compress and efficiently encode entity information (each node in the graph) which can later be used for several downstream tasks. Learning representations in an unsupervised fashion leads to task-agnostic representations that provide a general overview of a node's inherent characteristics, eliminating the need to re-train large networks to obtain node representations that may not translate well to other tasks. We focus on GraphSAGE, a popular unsupervised graph learning framework in our work, as it is empirically shown to have centrality-based biases.

3.1.2 GraphSAGE

GraphSAGE [Hamilton et al., 2017] works by *sampling and aggregating* information from the neighborhood of each node. The sampling component involves randomly sampling n-hop neighbors whose embeddings are then aggregated to update the node's own embedding. It works in the unsupervised setting by sampling a positive (nearby nodes) and a negative sample (distant nodes) for each node in the training batch. It then attempts: (a) to minimize the embedding distance between the node and the positive sample, and (b) to maximize the embedding distance between the node and the negative sample. The unsupervised version of GraphSAGE uses the following loss formulation,

$$\mathcal{J}_{\mathcal{G}}(z_u) = -\log(\sigma(z_u^T z_v)) - \mathcal{Q} \cdot \mathbb{E}_{v_n \sim P_n} \log(\sigma(-z_u^T z_{v_n})) \tag{3.1}$$

where v is a node that co-occurs near u on fixed-length random walk, σ is the sigmoid function, P_n is a negative sampling distribution, and Q defines the number of negative samples. z_u, z_v and z_{v_n} correspond to the learned embeddings of the training sample, the positive sample, and the negative sample, respectively. The loss landscape is formulated in such a way that it is minimized when z_u, z_v are close together and z_u, z_{v_n} are distant in the embedding space.

3.1.3 Graph Centrality Measures

Centrality measures correlate with a node's influence on a graph and capture the relative importance of nodes [Kang et al., 2011]. Among the several centrality measures that exist, we report results on degree centrality due to its popularity in current literature [Liu et al., 2021, Tang et al., 2020, Kang et al., 2022, Fish et al., 2019]. Another advantage is that the degree centrality for a graph can be calculated in linear time with respect to the number of edges, which is computationally inexpensive, unlike some of the other centralities.

3.1.4 Centrality-driven Group Fairness

Group Fairness concerns the disparity in the performance of a system for entities from different groups. Since most graph data does not possess explicit sensitive attributes, we utilize the connectivity

structure of the graph using centrality measures to naturally categorize nodes into groups - the group of popular (centrality greater than the median) and unpopular (centrality less than the median) nodes.

3.2 Fairness in GraphSAGE

GraphSAGE aggregates information from its neighbors, does not consider any intrinsic structural attributes, and focuses primarily on node attributes. Intrinsic graph structure information is very valuable irrespective of domain [Liu et al., 2021], and we believe that the learning process can be made fairer by leveraging aspects of this information. GraphSAGE takes the maximum number of nodes to be sampled from each hop neighborhood as a hyperparameter to build the computation graph, which could result in central nodes having complete and larger computation graphs while the less central ones having less information-rich subgraphs. As the size of the computation graph determines how much information the chosen node aggregates and learns from its neighborhood, the representations of central nodes encode much more information, giving them an advantage over less central nodes. Previous works [Kang et al., 2022] have theoretically and empirically proven the above claim for GCNs.

Theorem 1 *Suppose we have an input graph $\mathcal{G} = \{\mathcal{V}_\mathcal{G}, \mathbf{A}, \mathbf{X}\}$, the renormalized graph Laplacian $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\tilde{\mathbf{D}}^{-\frac{1}{2}}$, a nonlinear activation function σ and an L-layer GCN that minimizes a task-specific loss function J. For any l-th hidden graph convolution ($\forall l \in \{1, \ldots, L\}$) layer, the gradient of the loss function J with respect to the weight parameter $\mathbf{W}^{(l)}$ is a linear combination of the influence of each node weighted by its degree in the renormalized graph Laplacian.*

$$\frac{\partial J}{\partial \mathbf{W}^{(l)}} = \sum_{j=1}^{n} deg_{\hat{\mathbf{A}}}(j)\mathbf{I}_j^{(row)} = \sum_{i=1}^{n} deg_{\hat{\mathbf{A}}}(i)\mathbf{I}_i^{(col)}$$

where $deg_{\hat{\mathbf{A}}}(i)$ is the degree of node i in the renormalized graph Laplacian $\hat{\mathbf{A}}$,

$$\mathbf{I}_j^{(row)} = \left(\mathbf{H}^{(l-1)}[j,:]\right)^T \mathbb{E}_{i \sim p_{\hat{\mathcal{N}}(j)}}\left[\frac{\partial J}{\partial \mathbf{E}^{(l)}[i,:]}\right] \text{ and } \mathbf{I}_i^{(col)} = \left(\mathbb{E}_{j \sim p_{\hat{\mathcal{N}}(i)}}\left[\mathbf{H}^{(l-1)}[j,:]\right]\right)^T \frac{\partial J}{\partial \mathbf{E}^{(l)}[i,:]}$$

are the row-wise influence matrix of node j and the column-wise influence matrix of node i correspondingly. $\mathbf{H}^{(l-1)}$ is the input node embeddings of the hidden layer and $\mathbf{E}^{(l)} = \hat{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}$ is the node embeddings before the nonlinear activation.

Theorem 1, as proved in [Kang et al., 2022], implies that the node degree in $\mathbf{A}$ is proportional to its importance on the gradient of the weight matrix $\frac{\partial J}{\partial \mathbf{W}^{(l)}}$, implying that GCN is biased against low-degree nodes. This remark also stands true for our case as we use GraphSAGE with a mean aggregator (GraphSAGE-Mean behaves like a GCN [Hamilton et al., 2017]). As described in detail in the subsequent sections, we propose a fairer learning process, CAFIN, with higher penalties for less central nodes to tackle this issue. CAFIN helps prioritize the information flow in smaller computation graphs, as depicted in Figure 3.1, alleviating the disparity caused due to the computation graph sizes.

3.3 Imparity

We focus on inter-group fairness between groups constructed from intrinsic node characteristics rather than node features. The group membership is based on degree centrality in our results, however, CAFIN can be used with any centrality measure. From Figure 2, our initial analysis shows that GraphSAGE is biased towards nodes with more neighbors to learn from, and performs better for popular nodes. We use a modified variant of inter-group imparity [Kang et al., 2021] to measure the disparity in the performance of the embeddings between unpopular and popular nodes for different downstream tasks. Fairer representations would minimize the imparity between groups. Note that imparity is not used to train the network, but rather is exclusively an evaluation method.

3.3.1 Imparity for Node Classification

Node classification involves identifying labels for nodes in a graph [Tang et al., 2020, Hamilton et al., 2017]. As explained earlier, we divide the nodes into two groups (1 & 2) based on their centrality. We compute the inter-group accuracy differences for all classes weighted by the class distribution,

$$I_{nc} = \sum_{c \in C} w_c |a_1^c - a_2^c| \quad \text{and} \quad w_c = \frac{f_c}{|V|} \tag{3.2}$$

where I_{nc} represents the imparity for the task of node classification, f_c represents the count of nodes labeled with class c in the input graph, and a_i^c represents the average accuracy of nodes labeled with class c in the i^{th} group (either popular or unpopular nodes). We use a weighted metric (where the weights are proportional to the respective class cardinality) in place of the original [Kang et al., 2021] to avoid skewing the metric based on classes that are less common than others. In the case of *multi-label node classification* (such as in PPI dataset), we compute imparity as the difference in macro-F1 scores instead of accuracy, as it is a better representative of performance in the case of multi-label data [Grandini et al., 2020].

3.3.2 Imparity for Link Prediction

Link Prediction involves inferring whether an edge exists between two nodes solely from their attributes and local connectivity structure [Grover and Leskovec, 2016]. Based on our prior division of nodes based on popularity, edges are divided into three groups - between two popular nodes ($p - p$), between one popular and one unpopular node ($p - up$), and between two unpopular nodes ($up - up$). We define imparity as the standard deviation between the accuracies for these three types of edges.

$$I_{lp} = \sqrt{\frac{(a_{p-p} - \mu)^2 + (a_{p-up} - \mu)^2 + (a_{up-up} - \mu)^2}{3}} \tag{3.3}$$

$$\mu = \frac{a_{p-p} + a_{p-up} + a_{up-up}}{3} \tag{3.4}$$

where I_{lp} is the imparity for the task of link prediction and a_{x-y} is the accuracy of link prediction between nodes of type x and y. This formulation ensures that the metric is minimized when equal performance is observed across all three categories of edges. We choose standard deviation (SD) over the mean absolute deviation (MAD) to emphasize the effect of extreme outliers, better quantifying the overall fairness.

3.4 Preprocessing the Input Graph

Our proposed augmentations require pre-computed centrality measures for each node and the pairwise distances between all pairs of nodes. We pre-compute the pairwise distances using a breadth-first search (BFS) from each node while incrementally computing the degree centrality values simultaneously. Our framework utilizes the pairwise distances during training to impose the fairness constraints, while the centrality values are used later for defining group membership during evaluation.

Time complexity of this step can be broken down into three components. Let $|V|$ denote the number of nodes and $|E|$ denote the number of edges. Pairwise distance calculation uses BFS from each node and incurs $O(|V|^2 + |V||E|)$ in total but can easily be parallelized for improved performance. In section 4.5, we also explore efficient approximate distance measures as a potential replacement for this step to minimize the complexity, and we observe comparable results even with approximate distance measures. We calculate the degree centralities with one pass over all the edges, which incurs $O(|E|)$. We do not consider the centrality computation as overheads in our work, as they are used only to evaluate performance or to divide graphs that do not contribute to training time. The primary and most significant overhead is the pairwise distance computation which we consider a cost of fairness.

3.5 Preparing Graph Data for Inductive Setting

To translate transductive datasets to the inductive setting, we create disjoint subgraphs for each part of the pipeline. For both the downstream tasks (node classification and link prediction), we sample three subgraphs (g_1, g_2, and g_3) from the original graph: One for training GraphSAGE (g_1), one for training the downstream task classifier (g_2), and the other for evaluating the classifier's performance in the downstream task (g_3). We allocate more data for training GraphSAGE (g_1) than the downstream task classifier (g_2) as it has more parameters to learn.

Node Classification: Random vertex-induced subgraphs with 60% of the nodes for g_1, 30% for g_2 and the rest 10% for g_3.

Link Prediction: The subgraph for training embeddings g_1 is constructed by sampling 60% of the edges from the original graph. Since g_2 and g_3 deal with link prediction, they need positive samples (edges that actually exist) and negative samples (fabricated edges). We split the remaining edge set into g_{2p} and g_{3p} randomly (the positive edge set) and construct g_{2n} and g_{3n}, sets of artificial edges between

nodes that do not have an edge in the graph (the negative edge sets). The positive and negative edge partitions are merged to obtain the graphs g_2 and g_3, $g_2 = g_{2p} \cup g_{2n}$ and $g_3 = g_{3p} \cup g_{3n}$.

3.6 Centrality Aware Fairness Inducing In-processing (CAFIN)

We incorporate node degree and pairwise distance measures to augment GraphSAGE's loss formulation and achieve more equitable training between popular and unpopular nodes. Since this information is not dataset-specific, our method finds applications across domains without the inclusion of any dataset-specific overhead. Our proposed novel loss formulation is described below,

$$f_l(u, v) = \frac{\max_z \deg(z)}{\deg(u)} \cdot \log^2 \left(\frac{D(z_u, z_v)}{k} \cdot \frac{\max_{x,y}(d(x, y))}{d(u, v)} \right)$$

$$L_f = f_l(u, v) + f_l(u, v_n) \tag{3.5}$$

where $\deg(u)$ represents the degree of node u, z_u the embedding of node u, $D(z_u, z_v)$ the distance between the node embeddings of nodes u and v, and $d(u, v)$ the distance between the nodes in the graph. x, y, and z represent arbitrary nodes. Using GraphSAGE's notation, we represent the node of interest with u, the positive sample with v, and the negative sample with v_n. The parameter k normalizes the embedding distance and brings it to the same range as the normalized node distance. GraphSAGE's original loss formulation takes a contrastive form that we improve by considering the actual node distance. L_f, the final modified loss function, converges to 0 when the ratio between the two distances is 1, and our loss formulation tries to make the node embedding distance equal to the actual (normalized) distance between the nodes in the graph. Most real-world graphs are assortative in nature (similar nodes are close together) [Suresh et al., 2021]; hence, the actual node distance is a good proxy for the embedding distance.

We introduce a logarithm to curtail penalties for nodes whose embedding distances are distant from the actual node distances. Additionally, we square the overall formulation to ensure that the loss reaches a minimum when the embedding distance is equivalent to the actual node distance. The loss formulation focuses more on nodes with lower degrees, which conventionally have a lower impact on learning for GraphSAGE as they have few neighbors that they influence and are influenced by. Including the inverse of node degree helps shift focus toward less popular nodes, leading to less overall disparity during the learning process. We demonstrate that these enhancements lead to a fairer version of GraphSAGE on tasks that require node representations.

Intuition behind the loss function. Equation 3.5 details the loss formulation, and the goal is to make the actual distance in the graph directly proportional to the distance in the embedding space. Since the log function is naturally monotonic, we square the formulation to ensure that the minima occurs when the actual distance is equal to the embedding distance and the loss doesnt push the embedding distance to zero for all pairs of nodes (if we dont square the formulation, having an embedding distance of 0 would result in the least loss).

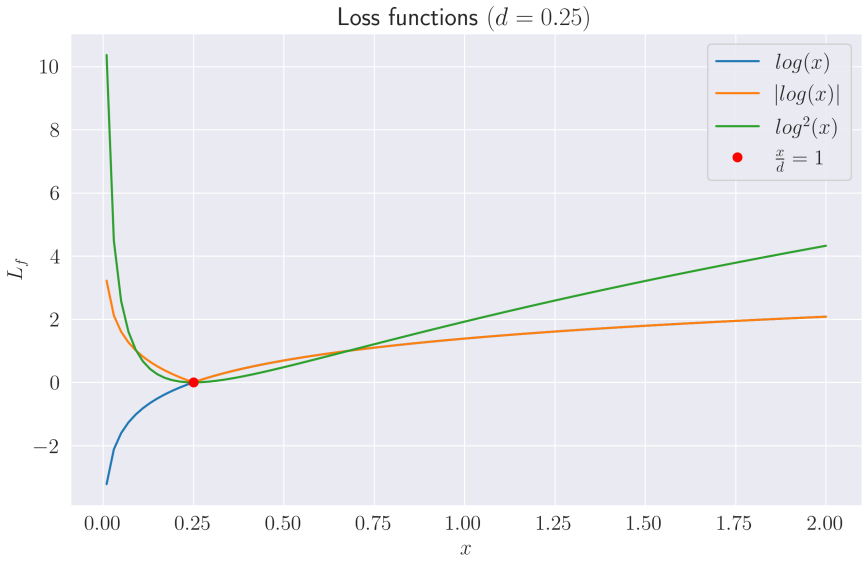

Figure 3.2 Loss Formulations - $log^2(x)$, a smooth function, achieves its minima at the required value

Considering x to be the embedding distance and d to be the actual node distance in the graph, we ideally want the minima to occur when $x = d$. We notice that this happens for both log^2 and $|log|$. As $|log|$ has a point of non-differentiability at $x = d$, and because smooth functions are preferred for gradient descent, we choose log^2 for our formulation. We have demonstrated the same in Figure 3.2.

Joint Training Strategy. To train CAFIN, we employ a joint training strategy that uses the original loss formulation as its primary objective and the modifications as its secondary.

$$L = L_o + \alpha L_f \tag{3.6}$$

Equation (3.6) describes the joint loss function where L_o is the original loss $\mathcal{J}_{\mathcal{G}}(z_u)$ described in equation (3.1), and L_f is the fairness-inducing constraint described in equation (3.5). α is a Lagrangian multiplier (balance factor) used to control the influence of the secondary fairness-inducing objective.

Stricter Sampling. GraphSAGE [Hamilton et al., 2017] requires positive and negative samples for each node for contrastive learning. They use a random node from the graph as the negative sample. We impose stricter constraints for selecting negative samples by leveraging the precomputed pairwise distances. We define a minimum distance threshold (NEG_MIN_DIST) for a node to qualify as a negative sample, i.e., the chosen negative sample must atleast be NEG_MIN_DIST hops away from the node ($d(u, v_n) \geq$ NEG_MIN_DIST). This ensures that the positive and negative samples are mutually exclusive, thus further improving the contrastive learning.

Chapter 4

Experiments with CAFIN

Here, we briefly describe the datasets we work with and the evaluation criteria we utilize. We then present the experimental results along with ablation studies.

Table 4.1 Dataset Description - We test CAFIN's efficacy over six datasets of various sizes from diverse domains.

Dataset	Nodes	Edges	Features	Classes
Cora	2,708	10,556	1,433	7
CiteSeer	3,327	9,104	3,703	6
Twitch	7,126	25,468	20	2
AMZN-P	7,650	238,162	745	8
AMZN-C	13,752	491,722	767	10
PPI	56,658 (24 graphs)	793,617	50	121 (multilbl.)

4.1 Datasets

We evaluate CAFIN on popular datasets spanning four domains, each possessing different network characteristics. Table 4.1 contains the quantitative description of each dataset.

Citation networks. We use the two most standard citation network benchmark datasets - Cora [McCallum et al., 2000], and CiteSeer [Giles et al., 1998]. In both the datasets, nodes correspond to manuscripts, and the undirected edges to citations. The node feature vector for both datasets is a word dictionary indicating the distribution of words in each paper's abstract, and the node label is the domain of the paper.

Social networks. We use the Twitch (EN) dataset [Rozemberczki et al., 2021] to study CAFIN's efficacy on social networks. The nodes in this dataset correspond to Twitch streamers, and the edges to mutual followers. Node features contain a representation of the games played by the streamer. The node label is binary and indicates if the streamer streams mature content.

E-commerce co-purchase networks. We use two datasets - Amazon Photos (AMZN-P) and Amazon Computers (AMZN-C) [Shchur et al., 2018]. In both datasets, the nodes correspond to products, and the edges connect co-purchased products. Node features contain the product reviews as bag-of-words, and the node label indicates the product category.

Protein interaction networks. Protein-Protein Interactions (PPI) [Zitnik and Leskovec, 2017] contains a collection of graphs (24), each an interaction network within different human tissues. Node features contain positional gene sets, motif gene sets, and immunological signatures, and the node label is a 121-dimensional vector corresponding to the gene ontology sets.

We use Cora [McCallum et al., 2000] and CiteSeer [Giles et al., 1998] from the citation network domain, Twitch (EN) [Rozemberczki et al., 2021] dataset to study CAFIN's efficacy on social networks, co-purchase networks - Amazon Photos (AMZN-P) and Amazon Computers (AMZN-C) [Shchur et al., 2018], and PPI [Zitnik and Leskovec, 2017] from the biological networks domain.

4.2 Evaluation Criteria

We evaluate the effective improvement in the model's fairness by comparing the change in imparity (refer to Section 3.3) with the original model. The lower the imparity value of an experiment, the fairer it is compared to the original. A decrease in the imparity value indicates the reduction of the model's performance disparity between the groups, depicted by a positive percentage in the tables. We also report the change in accuracy and the time overhead, the two primary costs of fairness for CAFIN.

Improvement in Imparity (II). The change in the imparity value measures the effective increase in fairness induced by the new formulations compared to the original. II measures the percentage decrease in imparity compared to the original. The higher the value of II, the fairer the formulation is. II is defined as,

$$II = \frac{I_o - I}{I_o} \cdot 100$$

where I_o corresponds to the imparity values of the original and I corresponds to the current imparity value.

Change in Accuracy (CA). Imposing fairness comes with a cost, like in most cases [Corbett-Davies et al., 2017, Menon and Williamson, 2018], generally in the form of a compromise in the model's performance. CA measures the overall model's accuracy change compared to the original. In an ideal experimental setting, CA will be close to 0. CA is defined as,

$$CA = A - A_o$$

where A_o corresponds to the overall accuracy of the original and A corresponds to the current overall accuracy.

Coefficient of Variance (CV). We measure the consistency of our results with the Coefficient of Variance (CV), which is defined as,

$$CV = \frac{\sigma}{\mu} \cdot 100$$

where μ corresponds the mean of observed results across runs and σ to the standard deviation. Low values of CV indicate consistency in the results. No specific ranges are considered acceptable in general as that depends on various factors like the experimental setting and objectives. [Aronhime et al., 2014] proposes that a CV value $\leq 10\%$ is considered excellent and anything between $10 - 20\%$ is considered good in their experimental setting.

Table 4.2 Results of CAFIN - (a) Link Prediction, II(CV) indicates a stable increase in fairness across datasets. (b) Node Classification, II indicates an increase in fairness, however, less consistently than link prediction (as indicated by the higher CV).

	(a) Link Prediction			(b) Node Classification		
Dataset	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**
Cora	20.48% (10.22%)	-2.75%	0.50	33.13% (10.86%)	0.19%	0.31
CiteSeer	62.89% (15.68%)	3.87%	0.18	17.71% (11.18%)	-1.00%	0.65
Twitch	38.92% (4.10%)	-5.67%	1.73	80.34% (12.06%)	-3.28%	0.84
AMZN-P	24.32% (5.26%)	-3.30%	4.80	32.63% (22.12%)	-7.70%	3.58
AMZN-C	53.07% (6.71%)	-4.56%	8.06	79.74% (38.30%)	-7.53%	5.37
PPI	73.31% (12.98%)	-3.28%	3.27	71.58% (3.28%)	-3.89%	3.35

Time Overhead per Increase in Imparity (T). This metric measures the effective increase in time per unit increase in imparity to give an idea about the effectiveness of the formulations with respect to the time overhead. Due to the augmentations, two parts in the pipeline could potentially incur a time overhead.

- *Training (t_t)* - We observe empirically that the time overhead in the training loop is insignificant in most cases. The increase for all datasets is less than 1% of the original time required to train, which is in milliseconds for 100 epochs. Nevertheless, for completeness, we add it to the final time overhead.
- *Preprocessing (t_p)* - The majority of the time overhead is constituted by preprocessing. We observe significant differences in this step as our augmentation requires extra information about the network to impose proposed constraints, specifically pairwise distance measures, which is an expensive operation. However, we propose a solution to this overhead in the form of approximate distance measures, later discussed in 4.5.2.

Based on the above two observations, we define T as,

$$T = \frac{t}{II}$$

where $t = t_t + t_p$ corresponds to CPU + I/O time (in seconds) required to precompute necessary data for the augmented formulation. We divide it by II to calculate the time spent to increase II by 1%. As the total time depends on various factors like the load on the hardware and other factors, we report the mean across 100 runs along with the CV. INF is reported when II is negative.

4.3 Results

The following results were obtained by using a linear SVM classifier for node classification, logistic regression for link prediction, and multiclass node classification (using a one vs. rest strategy), chosen due to their prevalence in the unsupervised learning paradigm, simplicity, and performance. The classifiers require the embeddings from GraphSAGE as input and use the train/test sets that were held out during the embedding training phase.

Table 4.2(a) captures our results for the link prediction task using all datasets and their respective fairness costs. We observe an improvement in imparity across the board with relatively low CV values, indicating stable improvements for this task. The drop in accuracy is reasonable across datasets and even positive in the case of Citeseer (indicating that in-processing for fairness can lead to performance enhancements in the case of extreme skews in centrality distribution). Although the time overhead is significant for larger graphs, we address it by including approximate distance measures, which results in minor reductions to both II and CA but a drastic reduction in T, which makes our method more feasible for large graphs.

Table 4.3 Results of Ablation studies for Link Prediction - (a) For CAFIN-N, the results are less consistent than the original. (b) For CAFIN-P, the results are less consistent than the original and, in some cases, much worse. (c) For CAFIN-AD, the approximations in pairwise node distances do not impact the II or CA by much, indicating the robustness of CAFIN and reassuring the scope of scalability.

	(a) CAFIN-N			(b) CAFIN-P		
Dataset	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**	**II ↑(CV ↓)**	**CA ↑**	**T ↓**
Cora	-12.98% (7.17%)	-3.56%	INF	-216.01% (1.10%)	-1.41%	INF
CiteSeer	84.66% (18.17%)	4.03%	0.13	42.26% (0.00%)	1.92%	0.26
Twitch	28.24% (5.35%)	-6.96%	2.34	48.87% (3.52%)	-10.04%	1.36
AMZN-P	21.24% (2.44%)	-3.87%	5.46	23.47% (0.82%)	-6.57%	5.00
AMZN-C	46.78% (9.20%)	-4.23%	9.14	49.66% (9.31%)	-7.03%	8.65
PPI	90.73% (26.14%)	-3.52%	2.65	76.05% (15.35%)	-3.01%	3.16

	(c) CAFIN-AD		
Dataset	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**
Cora	-12.68% (14.73%)	-5.47%	INF
CiteSeer	77.18% (13.01%)	-0.29%	0.07
Twitch	14.75% (5.70%)	-7.26%	0.34
AMZN-P	21.09% (5.73%)	-4.61%	0.24
AMZN-C	43.17% (4.46%)	-4.15%	3.59
PPI	24.46% (9.91%)	-2.96%	0.02

The low value of T, despite the size of the graph for PPI, is due to the distribution of its nodes and edges into multiple subgraphs, reducing the time overhead deviating from the trend of preprocessing time increasing with $|V|$ and $|E|$. The performance can be attributed to the inherent structure of the

dataset and how we exploit it to increase performance. The dataset comprises 24 disconnected subgraphs, each possessing a disjoint subset of the nodes in the graph. We perform pairwise distance computations on each subgraph individually and parallelly. It leads to a dramatic speedup as the pairwise distance computation is now proportional to $|V|$. The memory constraints are also relaxed as we split and store the pairwise distances for each component. Similar performance can be expected for any dataset with multiple connected components that can be processed in parallel.

Table 4.2(b) contains results for the node classification task, and we observe improvement in imparity for all datasets. We also observe that the improvements are much greater than the corresponding improvements in the link prediction task, with larger drops in accuracy. Although the improvement for the node classification task is better than that of link prediction, it is also more volatile than the improvement for link prediction. The lower variance in the link prediction task results stems from the task's simplicity when compared to node classification - binary classification compared to multiclass classification.

Table 4.4 Results of ablation studies for Node Classification - CAFIN-P and CAFIN-N show decreases in performance when compared to CAFIN. Even for cases where high II is observed, either the CV is very high or the CA is high. Results with * were run with AdaBoost classifier instead of LinearSVC.

	(a) CAFIN-N			(b) CAFIN-P		
Dataset	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**	**II ↑ (CV ↓)**	**CA ↑**	**T ↓**
Cora	35.23% (8.50%)*	-1.20%	0.29	-10.10% (7.44%)	-2.90%	INF
CiteSeer	75.61% (0.00%)	-15.43%	0.15	-55.48% (10.39%)	-6.28%	INF
Twitch	32.24% (51.86%)*	0.10%	2.05	10.57% (7.02%)	-1.76%	6.3
AMZN-P	55.37% (30.98%)*	-7.35%	2.09	16.92% (15.98%)	-17.90%	6.93
AMZN-C	88.83% (48.28%)*	-1.36%	4.82	42.54% (12.48%)	-2.94%	10.09
PPI	73.71% (4.70%)	-4.12%	3.26	6.29% (10.63%)	-1.05%	38.16

4.4 Hyperparameters

As GNNs are known to be sensitive to hyperparameters, we experiment with various combinations to obtain the best-performing values for each setting. The base configuration for training is 100 epochs, a learning rate of 0.0025, and a step learning rate scheduler. We tune the learning rate for each of the datasets that we do not detail in the interest of space. We use GraphSAGE with three layers and a hidden embedding size of 256 across runs and datasets. We experimented and empirically converged on $\alpha = 0.05$. All training runs were performed on an NVIDIA GeForce RTX 2080 Ti and 20 Intel Xeon E5-2640 v4 CPU cores with access to a minimum of 20GB of RAM.

GraphSAGE is known to be stochastic [Schumacher et al., 2020] due to the various random components it contains. Even then, CAFIN shows a consistent positive II (34.9% on average for the Twitch dataset) across popular seeds as seen in Figure 4.1. The values are averaged over 100 runs and we also

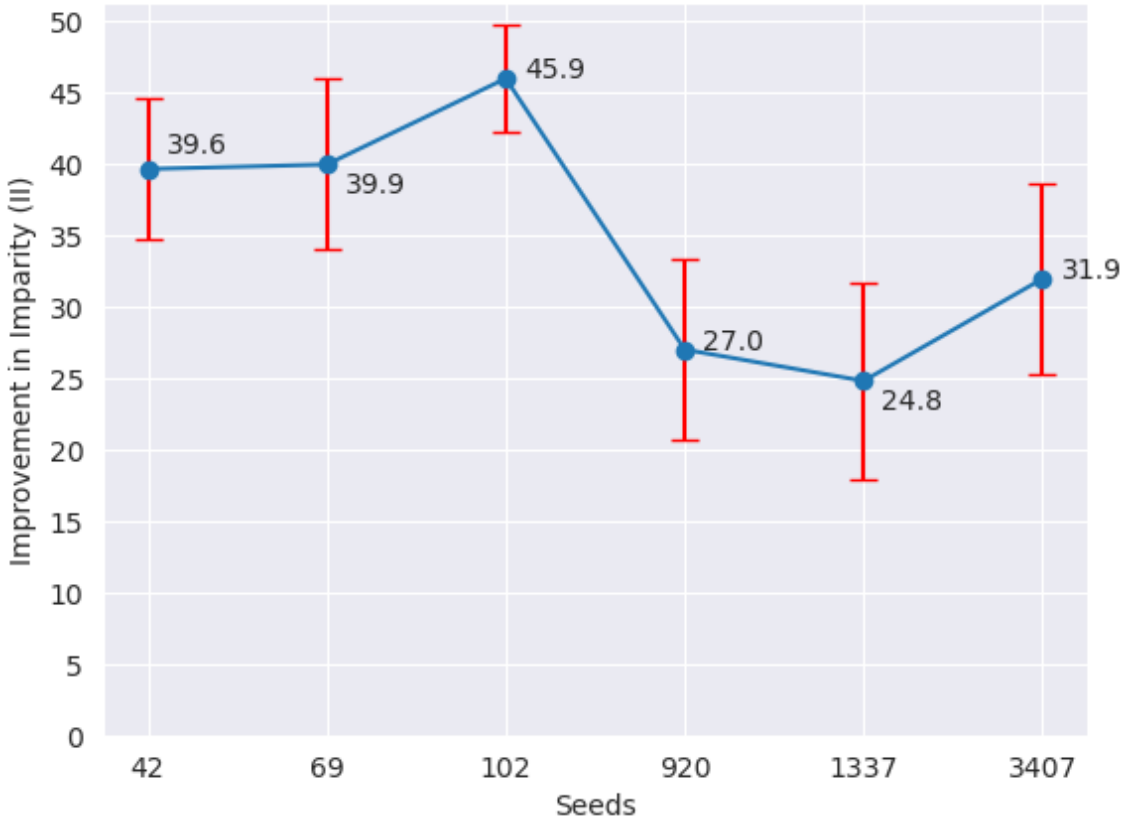

Figure 4.1 II vs popular seeds - Positive II is observed for various popular seeds (Avg - 34.9) for Link Prediction on the Twitch dataset.

plot the first standard deviation interval. Figure 4.1 shows that CAFIN is robust, and the results are reproducible across seeds, withstanding the stochastic nature of GraphSAGE.

4.5 Ablation Studies

We focus primarily on the loss formulation design to test which components of CAFIN lead to improvements and plausible solutions for the high time complexity of the dataset preprocessing step.

4.5.1 Loss Formulation Design

CAFIN treats positive and negative samples equally, but unpopular nodes have fewer positive samples than popular nodes, and the utilization of positive and negative samples may provide an unfair learning advantage to more popular nodes. To verify this theory, we construct two loss formulations based on the original hypothesis.

$$L_p(u, v) = f_l(u, v) \tag{4.1}$$

$$L_n(u, v_n) = f_l(u, v_n) \tag{4.2}$$

L_p (CAFIN-P) adds an additional term only for positive samples, and L_n (CAFIN-N) adds a term for only negative samples. The model is trained jointly, similar to Eq. 3.6 with the same parameter $\alpha = 0.05$. From tables 4.3(a) and 4.3(b), it can be observed that neither formulation performs consistently across datasets and either compromises on the improvement in imparity or the accuracy drop. Although we observe better performances for some datasets, CAFIN remains the preferred choice due to its stability. From Tables 4.4(a) and 4.4(b), we can observe that neither CAFIN-N nor CAFIN-P performs well consistently for Node Classification. It either compromises II, CA, or the robustness of the values. It is also important to note that the embeddings generated by CAFIN-N for Cora, AMZN-P, and AMZN-C were very homogenous. We used a more robust classifier with higher representative power, AdaBoost [Schapire, 1999], instead of our current one (LinearSVC) for the downstream task. Although we observe better performances for a couple of datasets, CAFIN is the preferred choice for this task as well, due to its balance of stability and performance.

Table 4.5 Results of t-test for Node Classification - CAFIN-P and CAFIN-N and CAFIN-P show a statistically significant change in distribution hence verifying the validity of our results.

Dataset	CAFIN-N	CAFIN-P
Cora	<0.00001	0.293251
CiteSeer	<0.00001	<0.00001
Twitch	<0.00001	<0.00001
AMZN-P	<0.00001	<0.00001
AMZN-C	<0.00001	0.361
PPI	<0.00001	<0.00001

We conducted a t-test for the statistical significance of the study results. We observe that in almost all the subject groups, the p-value is less than 0.00001. This indicates that the distributions are, in fact, independent of each other and that our ablation studies are statistically significant. Table 4.5 enumerates the p-values for CAFIN-N and CAFIN-P for every dataset (with respect to CAFIN) for the node classification task.

4.5.2 Approximate Distance Measures

CAFIN and its variants require an $O(|V|^2 + |V||E|)$ overhead to compute pairwise distances for the entire graph. This computation increases the time and space requirements during the preprocessing stage, inhibiting our in-processing technique's application to larger graphs. We demonstrate results using the landmark distance method to overcome this impediment [Potamias et al., 2009]. The landmark distance method considers several "landmarks" that are randomly chosen and computes the distance of every node to these landmarks during preprocessing. It utilizes the distance from landmarks and uses the triangle inequality to acquire an upper bound on the distance between any two nodes during inference. The landmark method for approximate distance measures reduces the time overhead from best

case $O(|V|^2)$ to $O(|V| \cdot l)$, where l is the number of landmarks chosen. We consider 100 landmarks for our experiments due to the performance-complexity trade-off. The time overhead can be approximated to a linear overhead for large graphs. Table 4.3(c) reports the results of CAFIN using landmark approximation (CAFIN-ApproximateDistance or CAFIN-AD) in place of the original pairwise distances. We observe a nominal drop in II compared to exact distance measures but a drastic reduction in the preprocessing time required, indicating that our method is robust to aberrations in distance measures.

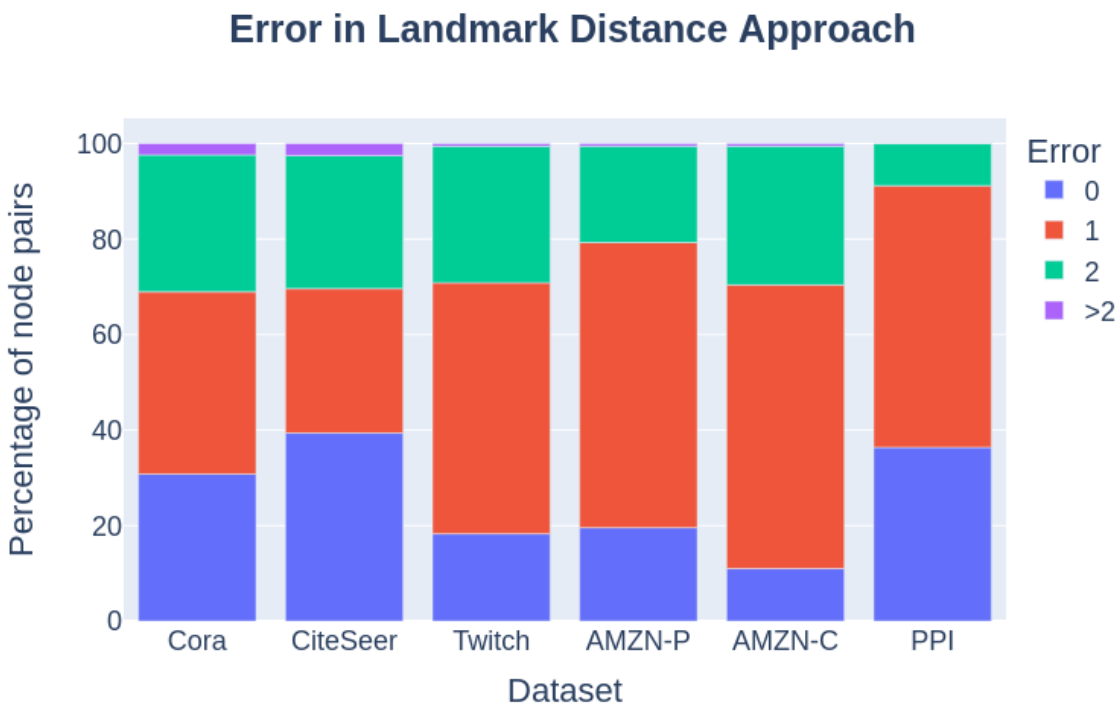

Figure 4.2 Error in Landmark-based Distance Approach - The figure illustrates the error of the Landmark Distance approach in calculating pairwise distances of nodes for various datasets. The color of the bars represents the error value in the predicted distance. An error of *x* denotes that the predicted distance deviates from the actual distance by *x*.

Landmark-based Approximate Distance Method Error Rates. CAFIN requires pairwise distances to operate and to impose a stricter sampling strategy; thus, it needs to be precomputed. The best time complexity (without approximations) for it is $O(|V|^2)$, where $|V|$ is the number of nodes in the graph. The time complexity becomes a bottleneck when we try to scale CAFIN to large graphs. To address this issue, we explore a fast approximate distance method.

As part of our ablation study, we try out *landmark-based* distance method for approximate distance computation [Potamias et al., 2009]. This method involves selecting a subset of nodes as "landmarks" and precomputing the distances of each node to those landmarks. When CAFIN requires the distance between a pair of nodes at runtime, it can be estimated quickly by combining the precomputed distances.

The new preprocessing step only requires computing the distance of each node to landmarks, which can be achieved by doing a breadth-first search from each landmark node. The time complexity for the same is $O(|V| \cdot l)$, where l is the number of landmark nodes.

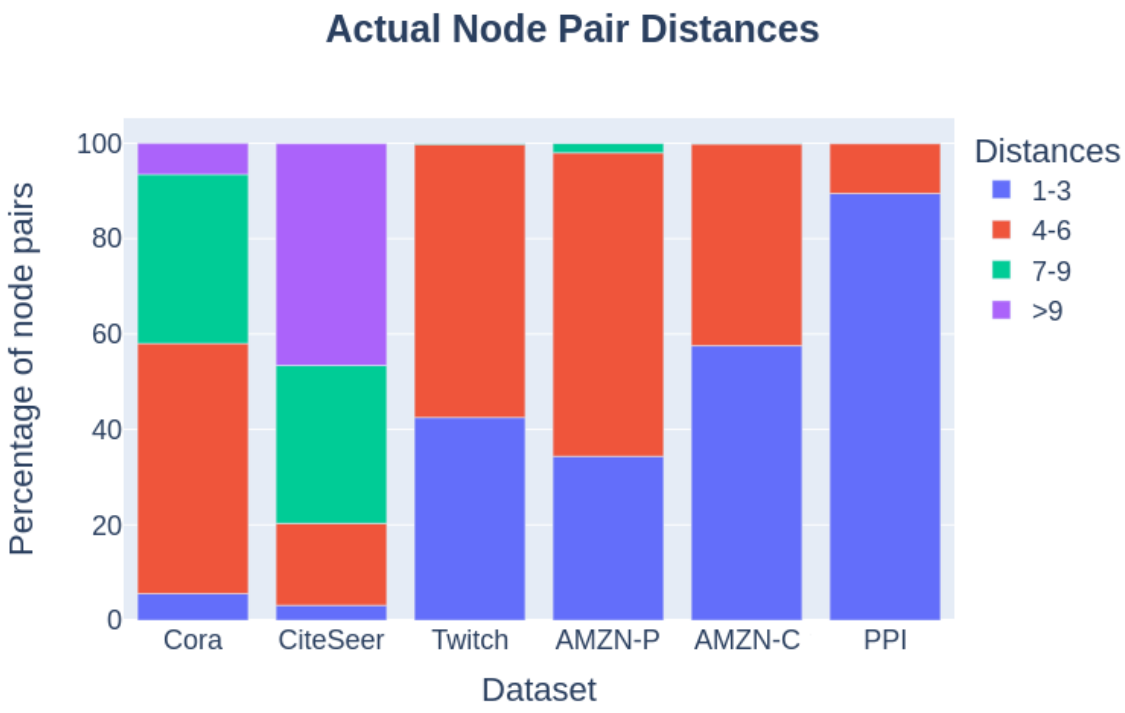

Figure 4.3 Actual Node Distances - The figure illustrates the correct node distances for each dataset. The bar's color represents the range the actual distance lies in (for example the blue part of the bar represents nodes with distances 1-3).

Let $d(u, v)$ be the actual distance between nodes u and v in the graph. It is important to note that the distance in graphs is a metric that satisfies the triangle inequality. That is, given any three nodes x, y, and z, the following inequality holds,

$$d(x, z) \leq d(x, y) + d(y, z) \tag{4.3}$$

$$d(x, z) \geq |d(x, y) - d(y, z)| \tag{4.4}$$

Note that if y belongs to one of the shortest paths from x to z, then the inequality 4.3 holds with equality. In order to compute the pairwise distances during runtime, we use the precomputed distances of nodes u and v from the landmark nodes and the triangle inequality. Let i be an arbitrary landmark node, and thus based on 4.3 and 4.4, for any two nodes u and v, we have,

$$\max_i |d(u, i) - d(v, i)| \leq d(u, v) \leq \min_i \{d(u, i) + d(i, v)\}$$

In other words, the true distance between nodes $d(u, v)$ lies in the range $[lb, ub]$, where $lb = \max_i |d(u, i) - d(v, i)|$ and $ub = \min_i\{d(u, i) + d(i, v)\}$. Any value in the range $[lb, ub]$ works as an approximation for $d(u, v)$. We use ub as an approximation as [Potamias et al., 2009] suggests that it is the best in most cases. Therefore, the complexity of computing distance during runtime for a pair of nodes is $O(l)$.

Figure 4.2 captures the absolute errors in the distance approximations across datasets. Figure 4.3 contains the graph's distribution of actual node distances. These two graphs show that the landmark method for distance approximation provides good approximations but has large error rates at times, further accentuating the robustness of CAFIN.

As expected, the performance of this method depends on the chosen subset of landmark nodes. A perfectly chosen subset can give accurate results for all distance pairs. Literature suggests picking nodes that have a high degree or high closeness centrality because they are more likely to be present in the shortest path of most pairs of nodes. As we aim to mitigate bias induced by a node's centrality, we do not use the above to maintain the pipeline bias-free. There is a trade-off between the performance and the computational complexity based on the number of landmarks chosen. More landmarks will guarantee stricter approximations but at the cost of more computation. Therefore, we select $l = 100$ landmark nodes for all datasets, which fared well on the trade-off through experimentation. Since we use the upper bound ub of triangle inequality to predict distance, our predicted distance can only be greater or equal to the actual distance. We increase the NEG_MIN_DIST value by 1 or 2 to account for the off by 1 or 2 error after noting the trend from Figure 4.2.

In the next section, we will move on to our second work on identifying dark patterns and fraud reviewer clusters on the Google Play Store.

Chapter 5

Dark Patterns in Install-Incentivizing Apps of Google Play[1]

Google Play lists over 2.89 million apps on its platform [Statista, 2022c]. In 2021 alone, these apps collectively accounted for over 111 billion installs by users worldwide [Statista, 2022a]. Given the magnitude of this scale, there is tremendous competition amongst developers to boost the visibility of their apps. As a result, developers spend considerable budgets on advertising, with expenditure reaching 96.4 billion USD on app installs in 2021 [Statista, 2022b]. Owing to this competitiveness, certain developers resort to inflating the reviews, ratings, and installs of their apps. The legitimacy of these means is determined by Google Plays policy, under which the use of incentivized installs is strictly forbidden [Google, 2022]. Some apps violate this policy by offering users incentives in the form of gift cards, coupons, and other monetary rewards in return for installing other apps; we refer to these as *install-incentivizing apps*. Past work [Farooqi et al., 2020] found that apps promoted on install-incentivizing apps are twice as likely to appear in the top charts and at least six times more likely to witness an increase in their install counts. While their work focuses on measuring the impact of incentivized installs on Google Play, our work aims to develop an understanding of how it affects the *users* of install-incentivizing apps. To this end, we perform a mixed-methods analysis of the reviews and permissions of install-incentivizing apps. Our ongoing work makes the following contributions:

1. We provide a detailed overview of various dark patterns present in install-incentivizing apps and highlight several normative concerns that disrupt the welfare of users on Google Play.

2. We examine different types of permissions requested by install-incentivizing apps to discover similarities with dark patterns, with 95% apps requesting permissions that access restricted data or perform restricted actions

[1]The following chapter is a part of *Ashwin Singh, Arvindh Arun, Pulak Malhotra, Pooja Desur, Ayushi Jain, Duen Horng Chau, and Ponnurangam Kumaraguru.* ***Erasing Labor with Labor: Dark Patterns and Lockstep Behaviors on Google Play.*** *In Proceedings of the 33rd ACM Conference on Hypertext and Social Media, HT 22, page 186 - 191, New York, NY, USA, 2022. Association for Computing Machinery.*

3. We show promising preliminary results in algorithmic detection of fraud and lockstep behaviors in reviews that boost the overall rating of install-incentivizing apps, detecting near-identical review pairs in 94% of the 50 most suspicious review clusters.

4. We release our dataset [Singh et al., 2022] comprising 319K reviews written by 301K reviewers over a period of five months and 1,825 most relevant reviews with corresponding qualitative codes across 60 install-incentivizing apps.

5.1 Dataset

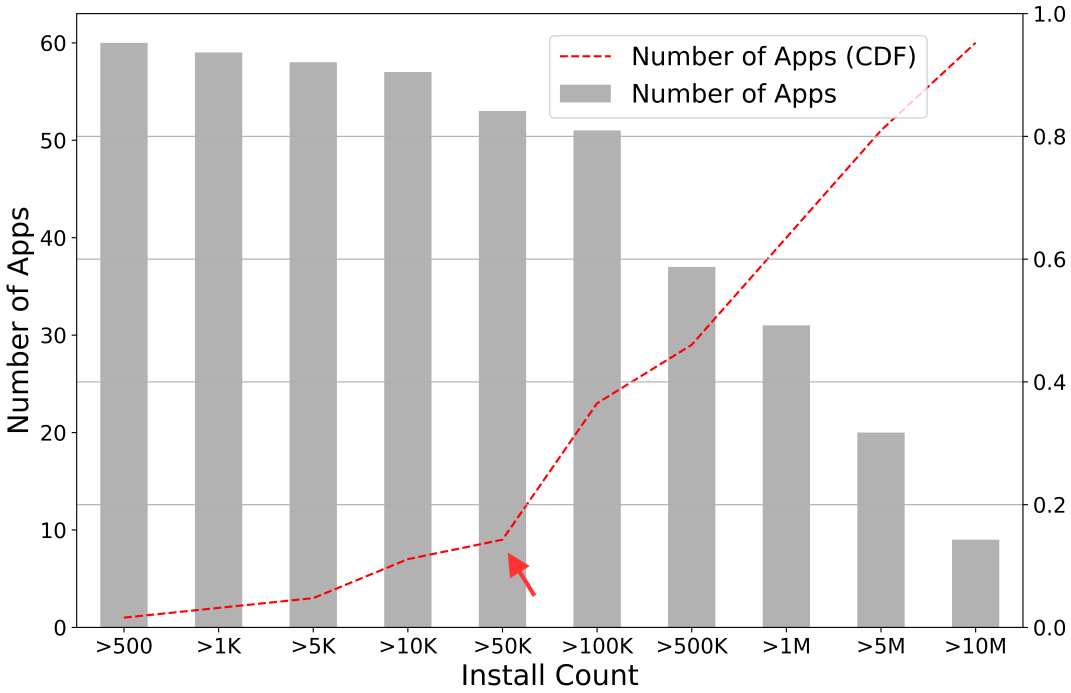

Figure 5.1 Distribution and CDF plot of install count for the 60 shortlisted install-incentivizing apps that collectively account for over 160.5M installs. Eighty-five percent of these apps have 100K or more installs, demonstrating their popularity.

We created queries by prefixing install apps with phrases like earn money, win prizes, win rewards, etc., and searched them on Google Play to curate a list of potentially install-incentivizing apps. Then, we proceeded to install the apps from this list on our mobile devices to manually verify whether these apps incentivized installs for other apps; we discarded the apps that did not fit this criterion. Following this process, we shortlisted 60 *install-incentivizing* apps. In Figure 5.1, we plot a distribution and CDF of their installs, finding that most apps (85%) have more than 100K installs. We used a scraper to collect reviews written daily on these apps, over a period of 5 months from November 1, 2021, to April 8, 2022. Reviews were collected daily to avoid over-sampling of reviews from certain temporal periods

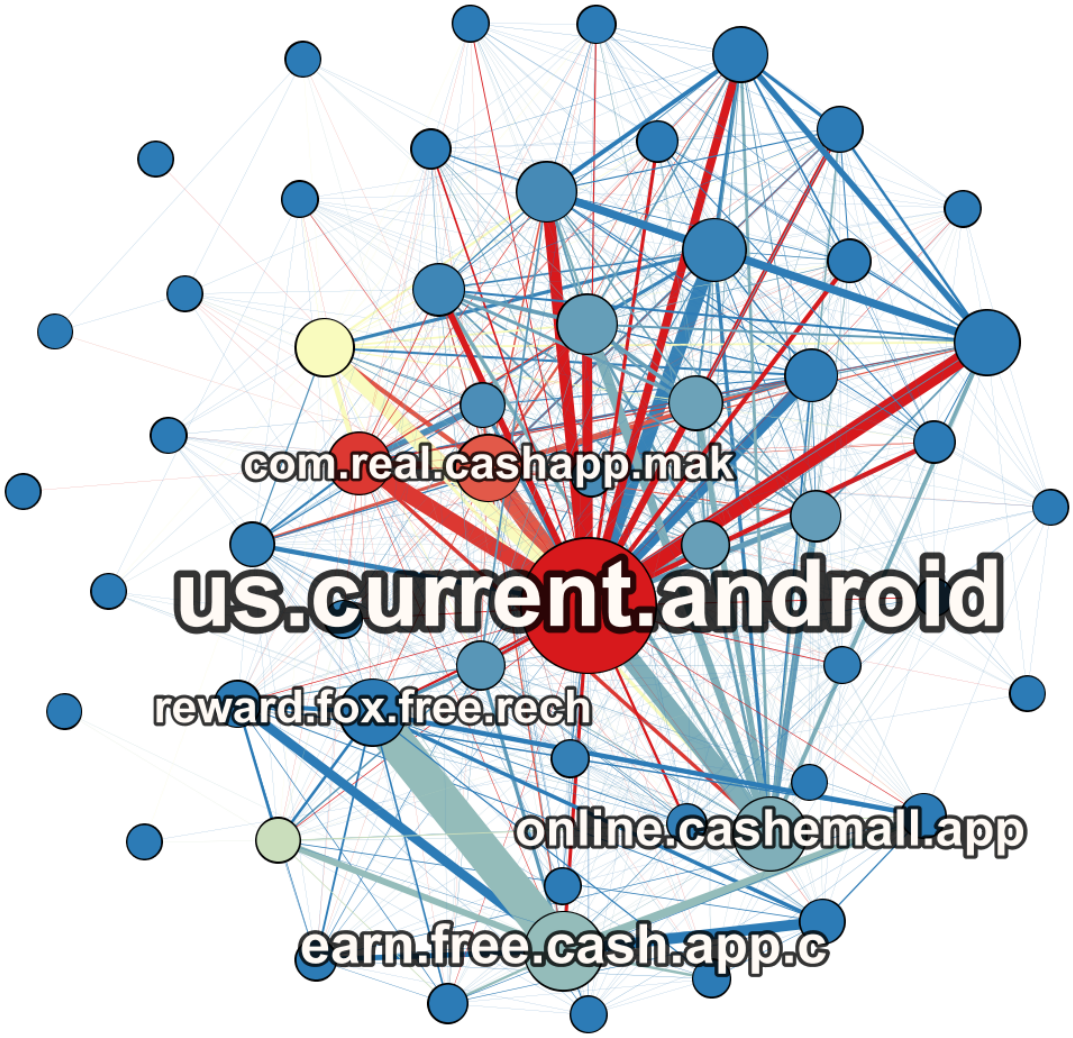

Figure 5.2 Network of apps showing labels of five apps that share the most reviewers with other apps. App 'us.current.android' shares 6.4K reviewers with other install-incentivizing apps.

over others. This resulted in 319,198 reviews from 301,188 reviewers. Figure 5.2 shows a network of apps where edges denote the number of reviewers shared by any two apps. We observe that certain apps share more reviewers with some apps over others, hinting at the possibility of collusion. Lastly, we also collected the permissions requested by apps on users devices.

5.2 Qualitative Findings

To understand the various ways in which install-incentivizing apps affect their users, we performed a qualitative analysis of their reviews. Unless a user expands the list of reviews, Google Play displays only the top four most relevant reviews under its apps. Owing to their default visibility, we sampled these

reviews for all 60 apps over a one-month period, obtaining 1,825 unique reviews. Then, we adopted an inductive open coding approach to thematically code [Miles and Huberman, 1994] these reviews. In the first iteration, all researchers independently worked on identifying high-level codes for these reviews which were then compared and discussed. During this process, we defined the 'completion of offers on install-incentivizing apps' as an act of *labor* by users and the 'incentive promised for their labor' as *value*. Then, we reached a consensus on four high-level themes: *exploitation, UI challenges, satisfaction*, and *promotion*, which we define below:

1. **Exploitation:** User invests *labor* but is unable to gain *value*.
2. **UI challenges:** User invests *labor* but the app's UI makes it challenging for them to gain *value*.
3. **Satisfaction:** User invests *labor* and is able to gain *value*.
4. **Promotion:** User invests *labor* in promoting an app through their review, rating or a referral code to gain *value*.

While all themes were useful for capturing the inter-relationship between a user's *labor* and its *value*, the first three themes were relatively more prevalent in our data. Next, we performed two iterations of line-by-line coding of reviews within the high-level themes where the researchers identified emerging patterns under each theme until the principle of saturation was established.

In this section, we describe our findings from the qualitative analysis to shed light on how install-incentivizing apps affect their users. More specifically, we elaborate on the commonalities and differences of patterns within high-level codes that we discovered using line-by-line coding to depict how labor invested by users in these apps is not only exploited but also leads to negative consequences for them as well as the platform.

5.2.1 Dark Patterns

Dark patterns can be defined as tricks embedded in apps that make users perform unintended actions [Brignull, 2018]. We find comprehensive descriptions of dark patterns present within install-incentivizing apps in reviews coded as 'exploitation' and 'UI challenges'. These patterns make it difficult for users to redeem value for their labor. First, our low-level codes uncover the different types of dark patterns present in reviews of install-incentivizing apps. Then, we ground these types in prior literature [Mathur et al., 2021] by utilizing lenses of both individual and collective welfare to highlight their normative concerns. The individual lens focuses on dark patterns that allow developers to benefit at the expense of users whereas the collective lens looks at users as a collective entity while examining expenses. In our case, the former comprises three normative concerns. First, patterns that enable developers to extract labor from users without compensating cause **financial loss (I1)** to users. Second, cases where the data of users is shared with third parties without prior consent, leading to **invasion of privacy**

Table 5.1 Different types of dark patterns mapped to their individual {Finanical Loss (I1), Invasion of Privacy (I2), Cognitive Burden (I3)} and collective {Competition (C1), Price Transparency (C2), Trust in the Market (C3)} normative concerns.

High-Level Code	Low-Level Code	Review	Normative Concerns					
			I1	I2	I3	C1	C2	C3
Exploitation	Withdrawal Limit	*100000 is equal to 10 dollars. Just a big waste of time. You can not reach the minimum cashout limit.*	✓			✓	✓	✓
	Cannot Redeem	*Absolute scam. Commit time and even made in app purchases to complete tasks ... I have over 89k points that it refuses to cash out!*	✓			✓	✓	✓
	Only Initial Payouts	*Good for the first one week then it will take forever to earn just a dollar. So now I quit this app ...*	✓			✓	✓	✓
	Paid Offers	*In the task I had to deposit 50 INR in an app and I would receive 150 INR as a reward in 24 hrs. 5 days have passed and I get no reply to mail.*	✓			✓	✓	✓
	Hidden Costs	*Most surveys say that the user isnt eligible for them, after you complete them! Keep in mind you may not be eligible for 90% of the surveys.*	✓			✓	✓	✓
	Privacy Violations	*Enter your phone number into this app and youll be FLOODED with spam texts and scams. I might have to change my phone number because I unwittingly ...*		✓				✓
UI Challenges	Too Many Ads	*Pathetic with the dam ads! Nothing but ads!!! Money is coming but only pocket change. Itll be 2022 before i reach $50 to cashout, if then.*			✓	✓		
	Progress Manipulation	*I redownload the app since the app would crash all the time ... I logged in and guess what?? ALL MY POINTS ARE GONE.. 12k points all gone...*	✓		✓		✓	✓
	Permission Override	*When you give it permission to go over other apps it* actually blocks everything else on your phone from *working correctly including Google to leave this review.*			✓	✓		✓

(I2). Third, when the information architecture of apps manipulates users into making certain choices due to the induced **cognitive burden (I3)**. The lens of collective welfare facilitates understanding of the bigger picture of install-incentivizing apps on Google Play by listing three additional concerns. Due to high **competition (C1)**, some developers incorporate dark patterns in apps that empower them to 'extract wealth and build market power at the expense of users [Day and Stemler, 2020] on the platform. In conjunction with their concerns at the individual level, they also pose a serious threat to the **price transparency (C2)** and **trust in the market (C3)** of Google Play. In Table 5.1, we show these different types of dark patterns mapped to their individual and collective normative concerns using sample reviews from our data.

5.2.2 Evidence of Fraudulent Reviews and Ratings

During qualitative analysis, we found that most reviews coded as 'satisfaction' were relatively shorter and lacked sufficient context to explain how the app benefitted the user, for e.g. *"Good app, "Nice App, "Very easy to buy money., "Nice app for earning voucher*. We performed Welchs *t*-test to validate that the number of words in reviews coded as satisfaction were very highly significantly lower than reviews coded as exploitation or UI challenges ($p < 0.001, t = -11.41$). The shorter length of reviews, along with the excessive use of adjectives and unrelatedness to the apps represented key spam-detection

signals [Shojaee et al., 2015], raising suspicions about their fraudulence. We discovered evidence of the same in reviews coded as 'promotion – *"Gets high rating because it rewards people to rate it so, "I rated it 5 stars to get credits*, thus finding that install-incentivizing apps also violate Google Plays policy by incentivizing users to boost their ratings and reviews. Other reviews coded as 'promotion involved users promoting other competitor apps (*"No earning 1 task complete not give my wallet not good ! CASHADDA App is good fast earning is good go install now thanks*) or posting their referral codes to get more credits within the install-incentivizing app (*'The app is Awesome. Use My Referral Code am****02 to get extra coin'*).

5.3 Quantitative Findings

In this section, we ascertain findings from our qualitative analysis as well as reveal more characteristics about the behavior of install-incentivizing apps and their reviews. For the same, we examine the permissions requested by these apps to establish their relevance to the dark patterns discussed in Section 5.2.1, and perform anomaly detection on their reviews to build upon the evidence of fraud from Section 5.2.2.

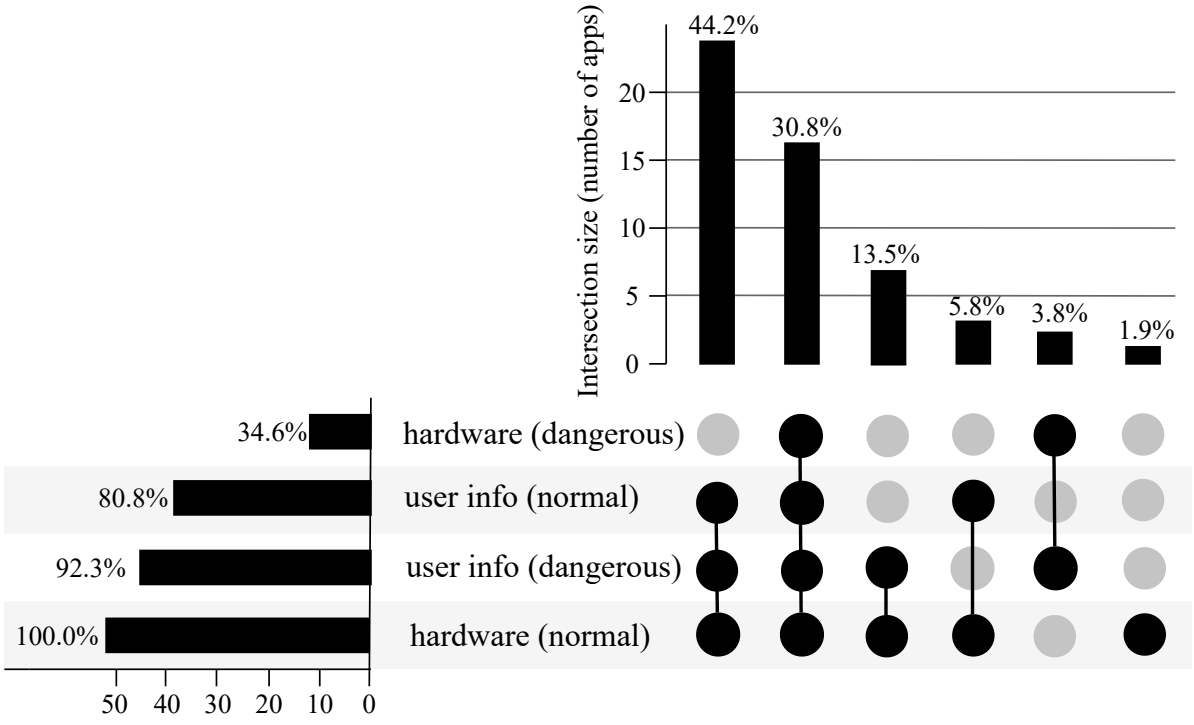

Figure 5.3 UpSet plot demonstrating different types of permissions present in install-incentivizing apps. Over ninety-two percent of apps request permissions that access sensitive user information.

5.3.1 Permissions in Install-Incentivizing Apps

App permissions support user privacy by protecting access to restricted data and restricted actions on a users device [Developers, 2022]. Most permissions fall into two protection levels as determined by Android, namely *normal* and *dangerous*, based on the risk posed to user privacy. Similarly, another distinction can be made between permissions that access *user information* and permissions that only *control device hardware* [Center, 2015]. We leverage these categories in our analysis to identify types of permissions prominent across install-incentivizing apps. Figure 5.3 shows an UpSet plot [Lex et al., 2014] of different types of permissions present in install-incentivizing apps. First, we observe that over 92% of apps comprise *dangerous* permissions that access user information. The most popular permissions in this category include 'modify or delete the contents of your USB storage (41 apps), 'read phone status and identity (24 apps), 'access precise location (19 apps), and 'take pictures and videos (14 apps). Second, despite being requested by relatively fewer apps, some permissions in this category enable an alarming degree of control over user information; for e.g. 'create accounts and set passwords (5 apps), 'add or modify calendar events and send email to guests without owners' knowledge (3 apps) and 'read your contacts (2 apps). Third, 34% of install-incentivizing apps contain permissions that access dangerous hardware-level information, the most prominent one being 'draw over other apps (14 apps). Fourth, we note that all but three apps request at least one dangerous permission. Lastly, permissions requested by install-incentivizing apps share common characteristics with the dark patterns discussed above, thus validating their qualitative discovery.

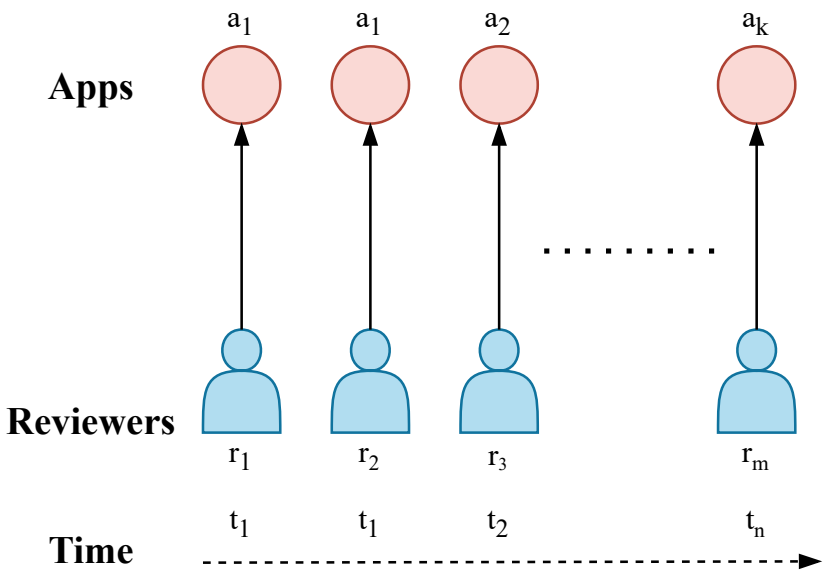

Figure 5.4 Reviews are modeled as an edge-stream in a dynamic bipartite graph of apps and reviewers. Each edge $e \in E$ represents a tuple (r, a, t) where r is a reviewer who reviews an app a at time t.

5.3.2 Lockstep Behaviors

In Section 5.2.2, we found evidence of install-incentivizing apps indulging in review and rating fraud. Thus, we build upon the same to investigate reviews of these apps for anomalous behaviors such as lockstep that are indicative of fraud. Specifically, we focus on detecting groups of reviews that exhibit similar temporal and rating patterns; for e.g. bursts of reviews on an app within a short period of time to boost its overall rating.

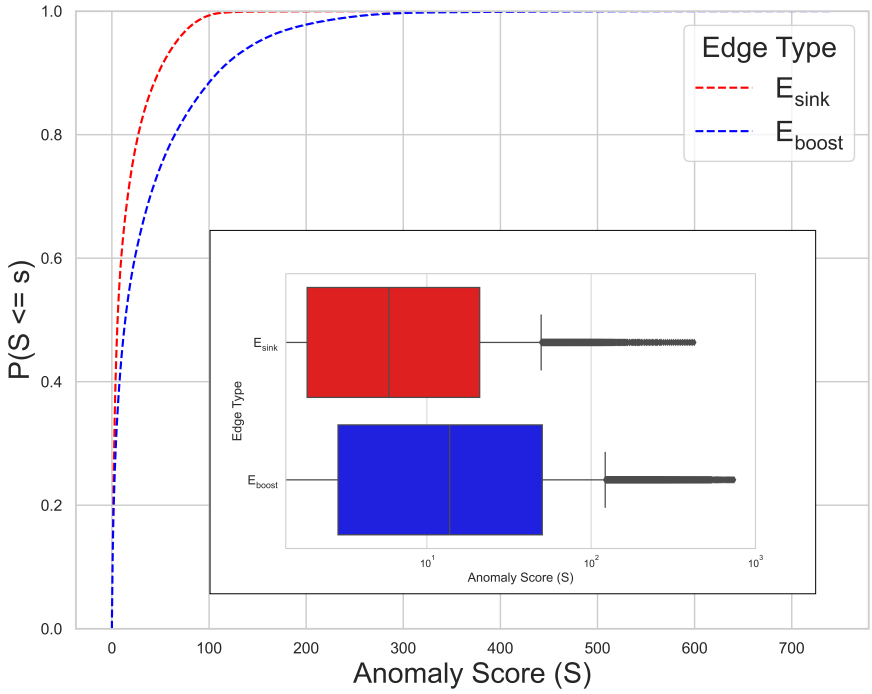

Figure 5.5 CDF plot of anomaly scores for the two edge streams E_{boost} and E_{sink}. Reviews that boost the overall rating of an install incentivizing app exhibit significantly more anomalous behavior than reviews that aim to bring it down.

5.3.2.1 Modelling and Experimental Setup

Given that reviews are a temporal phenomenon, we model them as an edge-stream $E = \{e_1, e_2, ...\}$ of a dynamic graph G. Each edge $e_i \in E$ represents a tuple (r_i, a_i, t_i) where r_i is a reviewer who reviews an app a_i at time t_i (see Fig 5.4). Groups of fraudulent reviewers may either aim to boost the overall rating of an install-incentivizing app or sink the rating of a competitor app. Thus, we partition our edge stream into two sub-streams as follows:

1. $E_{boost} = \{(r_i, a_i, t_i) \in E \mid \text{Score}(r_i, a_i) \geq R_{a_i}\}$, $|E_{boost}| = 215,759$
2. $E_{sink} = \{(r_i, a_i, t_i) \in E \mid \text{Score}(r_i, a_i) < R_{a_i}\}$, $|E_{sink}| = 103,439$

where Score$(r_i, a_i) \in \{1, 2, 3, 4, 5\}$ is the score assigned by reviewer r_i to the app a_i and R_{a_i} denotes the overall rating of app a_i. Next, we reconfigure a state-of-the-art microcluster anomaly detection algorithm MIDAS-F [Bhatia et al., 2022] for our use. In particular, we modify the definition of a microcluster to accommodate the bipartite nature of our dynamic graph. Given an edge $e \in E$, a detection period $T \geq 1$ and a threshold $\beta > 1$, there exists a microcluster of reviews on an app a if it satisfies the following equation:

$$\frac{c(e, (n+1)T)}{c(e, nT)} > \beta \text{ where } c(e, nT) = \left|\{(r_i, a, t_i) \mid (r_i, a, t_i) \in E_{boost} \wedge (n-1)T < t_i \leq nT\}\right| \quad (5.1)$$

if $e \in E_{boost}$ and vice versa for E_{sink}. Depending on whether e is a boosting or sinking edge, $c(e, nT)$ counts similar edges for the app a within consecutive detection periods $(n-1)T$ and nT. Values recommended by the authors are used for the remaining parameters α and θ. It is worth noting that our modification preserves its properties of (i) theoretical guarantees on false positive probability, and (ii) constant-time and constant-memory processing of new edges [Bhatia et al., 2022].

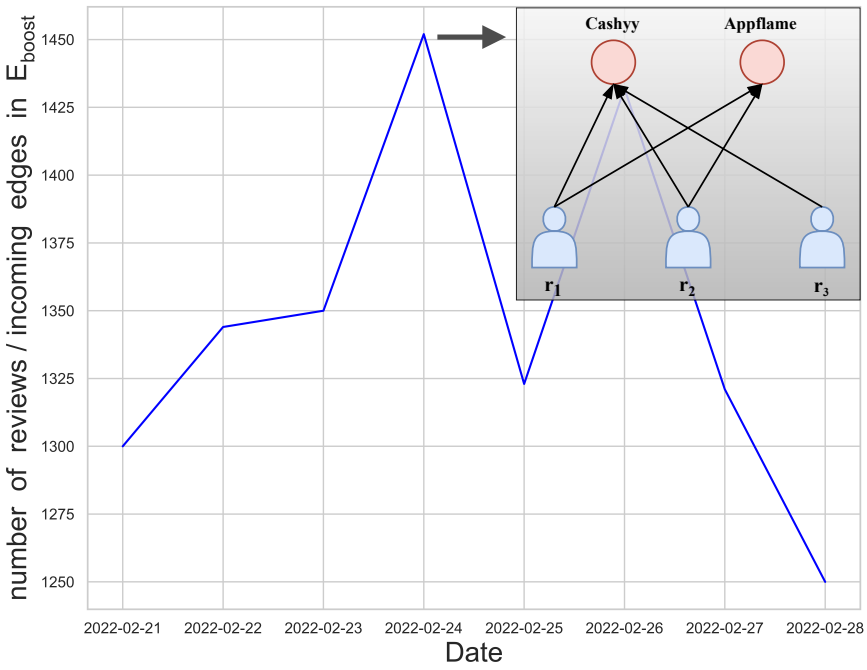

Figure 5.6 A microcluster anomaly detected by the algorithm. Three reviewers are boosting the overall rating of two install-incentivizing apps 'Cashyy' and 'Appflame' on the same day.

5.3.2.2 Analysis and Preliminary Results

MIDAS-F follows a streaming hypothesis testing approach that determines whether the observed and

expected mean number of edges for a node at a given timestep are significantly different. Based on a chi-squared goodness-of-fit test, the algorithm provides anomaly scores $\mathrm{S}(e)$ for each edge e in a streaming setting. Upon computing anomaly scores for both sub-streams E_{boost} and E_{sink}, we visualize their CDF with an inset box plot in Fig 5.5. It can be observed that E_{boost} exhibits more anomalous behavior than E_{sink}. To ascertain the statistical significance of the same, we make use of Welch's t-test for the hypothesis $H_1 : \mathrm{S}_\mu(E_{boost}) > \mathrm{S}_\mu(E_{sink})$. We infer that reviews that aim to boost the rating of an install-incentivizing app show anomalous behavior that is highly significantly more ($t = 157.23, p < 0.0$) than reviews that aim to bring it down.

Next, we examine fraud across anomalous microclusters detected by the algorithm. Figure 5.6 shows one such microcluster anomaly where the algorithm detects reviews from three reviewers boosting the overall rating of two install-incentivizing apps on the same day. We extract the 50 most suspicious clusters of reviews from both sub-streams E_{boost} and E_{sink} based on their average anomaly scores. For each pair of reviews (r_i, r_j) within these clusters, we compute their cosine similarity $CS(r_i, r_j)$ using embeddings generated by Sentence-BERT [Reimers and Gurevych, 2019]. Over 35% of reviews (1,687 of 4,717) from the suspicious clusters in E_{boost} form at least one pair of highly identical reviews i.e., $CS(r_i, r_j) = 1$. However, this percentage drops to 10% (45 of 432 reviews) in the case of E_{sink}. On closer inspection, we find that these are all extremely short reviews with at most three to four words that comprise mostly of adjectives; for e.g., E_{boost}: ('good app', 'very good app'), ('good earning app', 'very good for earning app'), ('best app', 'very best app') and E_{sink}: ('bad', 'very bad'), ('super', 'super'), ('nice', 'very nice'). It is surprising to see that all but four identical pairs from E_{sink} contain only positive adjectives considering they assign the app a low rating. A potential reason for this dissonance can be that reviewers writing these reviews want to camouflage as normal users in terms of their rating patterns. Lastly, from the fifty most suspicious clusters, we find such pairs across 47 (94%) clusters from E_{boost} and 21 (42%) clusters from E_{sink}. This demonstrates that the efficacy of our approach towards detecting lockstep behaviors is not only limited to the temporal and rating dimensions but also extends to the content present in reviews.

Chapter 6

Conclusion, Limitations and Future Work

We introduce CAFIN, a fairness-inducing in-processing technique, and demonstrate its efficacy in reducing degree-based disparities in the embeddings generated by GraphSAGE. CAFIN offers an average of 49.50% and 52.52% improvement in Imparity for Link Prediction and Node Classification tasks, respectively, across datasets. We test CAFIN's robustness by conducting various ablation studies. We also introduce the CAFIN-AD variant, which uses approximate distances for reduced computational complexity, making it highly scalable and deployable in more extensive settings. We believe that CAFIN can be extended to any other contrastive-learning-based framework in this domain..

Our second work sheds light on how lax implementation of Google Plays policy on fraudulent installs, ratings, and reviews empowers developers of install-incentivizing apps to deplete the trust and transparency of the platform. Through the use of permissions that access restricted data and perform restricted actions, developers incorporate dark patterns in these apps to deceive users and extort labor from them in the form of offers. The second form of labor that we study in our work is the writing of fraudulent reviews. We find evidence of their presence qualitatively and show promising results in detecting them algorithmically. Both types of fraud (incentivized installs and reviews) are only made possible by the labor of users who are vulnerable or crowd-workers who are underpaid [Rahman et al., 2019]. This enables developers to extract profits as they get away with violating Google Plays policies without any consequences or accountability.

CAFIN fills a niche, but a crucial gap in the centrality-driven fairness paradigm, so its focus is targeted. Interpretability and explainability of graph learning algorithms also need to be explored - and this work does not seek to address these concerns. Further, CAFIN does not compare and contrast the impact of various centrality measures on the fairness constraints, as it imposes only degree-centrality-based fairness constraints, it is another direction to explore. We hope our work promotes further investigation in the domain of fairness for unsupervised GNNs, explicitly focusing on graph structure-induced biases.

In the context of identifying microclusters of anomalous users, a question that remains unanswered is, if reviews under these apps describe exploitative experiences of users, what is it that facilitates their continued exploitation? For now, we can only conjecture that fraudulent positive reviews on install-

incentivizing apps suppress ranks of reviews containing exploitative experiences of users. Whether the same holds true or not is a question that remains to be explored in our future work.

With these works, we hope that we have demonstrated the need and the scope for pushing toward a more equitable and fair online environment by both ensuring that models don't carry any form of biases and also by constantly detecting and addressing harmful user behavior.

Related Publications

- *Arvindh Arun, Aakash Aanegola, Amul Agrawal, Ramasuri Narayanam, and Ponnurangam Kumaraguru.* **CAFIN: Centrality Aware Fairness Inducing IN-processing for unsupervised representation learning on graphs.** In ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Krakow, Poland, volume 372 of Frontiers in Artificial Intelligence and Applications, pages 101 - 108. IOS Press, 2023.

- *Ashwin Singh, Arvindh Arun, Pulak Malhotra, Pooja Desur, Ayushi Jain, Duen Horng Chau, and Ponnurangam Kumaraguru*. **Erasing Labor with Labor: Dark Patterns and Lockstep Behaviors on Google Play.** In Proceedings of the 33rd ACM Conference on Hypertext and Social Media, HT 22, page 186 - 191, New York, NY, USA, 2022. Association for Computing Machinery.

Other Publications

- *T. H. Arjun, Arvindh A., and Kumaraguru Ponnurangam.* Precog-LTRC-IIITH at GermEval 2021: **Ensembling pre-trained language models with feature engineering**. In Proceedings of the GermEval 2021 Shared Task on the Identification of Toxic, Engaging, and Fact-Claiming Comments, pages 3946, Duesseldorf, Germany, September 2021. Association for Computational Linguistics.

Bibliography

[Aronhime et al., 2014] Aronhime, S., Calcagno, C., Jajamovich, G. H., Dyvorne, H. A., Robson, P., Dieterich, D., Isabel Fiel, M., Martel-Laferriere, V., Chatterji, M., Rusinek, H., and Taouli, B. (2014). Dce-mri of the liver: Effect of linear and nonlinear conversions on hepatic perfusion quantification and reproducibility. *Journal of Magnetic Resonance Imaging*, 40(1):90–98.

[Avin et al., 2015] Avin, C., Keller, B., Lotker, Z., Mathieu, C., Peleg, D., and Pignolet, Y.-A. (2015). Homophily and the glass ceiling effect in social networks. In *Proceedings of the 2015 conference on innovations in theoretical computer science*, pages 41–50.

[Bertrand and Mullainathan, 2004] Bertrand, M. and Mullainathan, S. (2004). Are emily and greg more employable than lakisha and jamal? a field experiment on labor market discrimination. *American economic review*, 94(4):991–1013.

[Bhatia et al., 2022] Bhatia, S., Liu, R., Hooi, B., Yoon, M., Shin, K., and Faloutsos, C. (2022). Real-time anomaly detection in edge streams. *ACM Trans. Knowl. Discov. Data*, 16(4).

[Brignull, 2018] Brignull, H. (2018). Deceptive designs. `https://www.deceptive.design/`.

[Buet-Golfouse and Utyagulov, 2022] Buet-Golfouse, F. and Utyagulov, I. (2022). Towards fair unsupervised learning. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 13991409, New York, NY, USA. Association for Computing Machinery.

[Center, 2015] Center, P. R. (2015). An analysis of android app permissions. `https://www.pewresearch.org/internet/2015/11/10/an-analysis-of-android-app-permissions/`.

[Choudhary et al., 2022] Choudhary, M., Laclau, C., and Largeron, C. (2022). A survey on fairness for machine learning on graphs. *arXiv preprint arXiv:2205.05396*.

[Corbett-Davies et al., 2017] Corbett-Davies, S., Pierson, E., Feller, A., Goel, S., and Huq, A. (2017). Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 797806, New York, NY, USA. Association for Computing Machinery.

[Day and Stemler, 2020] Day, G. and Stemler, A. (2020). Are dark patterns anticompetitive? *Ala. L. Rev.*, 72:1.

[Developers, 2022] Developers, A. (2022). Permissions on android. `https://developer.android.com/guide/topics/permissions/overview`.

[Dwork et al., 2012] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226.

[Farooqi et al., 2020] Farooqi, S., Feal, A., Lauinger, T., McCoy, D., Shafiq, Z., and Vallina-Rodriguez, N. (2020). Understanding incentivized mobile app installs on google play store. In *Proceedings of the ACM Internet Measurement Conference*, IMC '20, page 696709.

[Fish et al., 2019] Fish, B., Bashardoust, A., Boyd, D., Friedler, S., Scheidegger, C., and Venkatasubramanian, S. (2019). Gaps in information access in social networks? In *The World Wide Web Conference*, WWW '19, page 480490, New York, NY, USA. Association for Computing Machinery.

[Giles et al., 1998] Giles, C. L., Bollacker, K. D., and Lawrence, S. (1998). Citeseer: an automatic citation indexing system. In *Digital library*.

[Google, 2022] Google (2022). User ratings, reviews, and installs. `https://support.google.com/googleplay/android-developer/answer/9898684`.

[Grandini et al., 2020] Grandini, M., Bagli, E., and Visani, G. (2020). Metrics for multi-class classification: an overview.

[Grover and Leskovec, 2016] Grover, A. and Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. In *Proceedings of KDD*, pages 3541–3551.

[Hamilton et al., 2017] Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 10251035, Red Hook, NY, USA. Curran Associates Inc.

[Hardt et al., 2016a] Hardt, M., Price, E., Price, E., and Srebro, N. (2016a). Equality of opportunity in supervised learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

[Hardt et al., 2016b] Hardt, M., Price, E., and Srebro, N. (2016b). Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29.

[Kang et al., 2021] Kang, J., Xie, T., Wu, X., Maciejewski, R., and Tong, H. (2021). Infofair: Information-theoretic intersectional fairness.

[Kang et al., 2022] Kang, J., Zhu, Y., Xia, Y., Luo, J., and Tong, H. (2022). Rawlsgcn: Towards rawlsian difference principle on graph convolutional network. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 12141225, New York, NY, USA. Association for Computing Machinery.

[Kang et al., 2011] Kang, U., Papadimitriou, S., Sun, J., and Tong, H. (2011). *Centralities in Large Networks: Algorithms and Observations*, pages 119–130.

[Karimi et al., 2018] Karimi, F., Génois, M., Wagner, C., Singer, P., and Strohmaier, M. (2018). Homophily influences ranking of minorities in social networks. *Scientific reports*, 8(1):1–12.

[Khajehnejad et al., 2020] Khajehnejad, M., Rezaei, A. A., Babaei, M., Hoffmann, J., Jalili, M., and Weller, A. (2020). Adversarial graph embeddings for fair influence maximization over social networks. *arXiv preprint arXiv:2005.04074*.

[Krasanakis et al., 2021] Krasanakis, E., Papadopoulos, S., and Kompatsiaris, I. (2021). Applying fairness constraints on graph node ranks under personalization bias. In *International Conference on Complex Networks and Their Applications*, pages 610–622. Springer.

[Leskovec et al., 2020] Leskovec, J., Rajaraman, A., and Ullman, J. D. (2020). *Mining of Massive Datasets*. Cambridge University Press. `http://mmds.org`.

[Lex et al., 2014] Lex, A., Gehlenborg, N., Strobelt, H., Vuillemot, R., and Pfister, H. (2014). Upset: Visualization of intersecting sets. *IEEE Transactions on Visualization and Computer Graphics (InfoVis)*, 20(12):1983–1992.

[Liu et al., 2020] Liu, J., Ong, G. P., and Chen, X. (2020). Graphsage-based traffic speed forecasting for segment network with sparse data. *IEEE Transactions on Intelligent Transportation Systems*.

[Liu et al., 2021] Liu, Z., Nguyen, T.-K., and Fang, Y. (2021). Tail-gnn: Tail-node graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21, page 11091119, New York, NY, USA. Association for Computing Machinery.

[Liu et al., 2023] Liu, Z., Nguyen, T.-K., and Fang, Y. (2023). On generalized degree fairness in graph neural networks.

[Lo et al., 2022] Lo, W. W., Layeghy, S., Sarhan, M., Gallagher, M., and Portmann, M. (2022). E-graphsage: A graph neural network based intrusion detection system for iot. In *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9. IEEE.

[Mathur et al., 2019] Mathur, A., Acar, G., Friedman, M. J., Lucherini, E., Mayer, J., Chetty, M., and Narayanan, A. (2019). Dark patterns at scale: Findings from a crawl of 11k shopping websites. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW).

[Mathur et al., 2021] Mathur, A., Kshirsagar, M., and Mayer, J. (2021). What makes a dark pattern... dark? design attributes, normative considerations, and measurement methods. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21.

[McCallum et al., 2000] McCallum, A., Nigam, K., Rennie, J. D. M., and Seymore, K. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163.

[Mehrabi et al., 2021] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6).

[Menon and Williamson, 2018] Menon, A. K. and Williamson, R. C. (2018). The cost of fairness in binary classification. In Friedler, S. A. and Wilson, C., editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 107–118. PMLR.

[Mildner et al., 2023] Mildner, T., Savino, G.-L., Doyle, P. R., Cowan, B. R., and Malaka, R. (2023). About engaging and governing strategies: A thematic analysis of dark patterns in social networking services. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA. Association for Computing Machinery.

[Miles and Huberman, 1994] Miles, M. B. and Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. sage.

[Newman, 2003] Newman, M. E. (2003). Mixing patterns in networks. *Physical review E*, 67(2):026126.

[O'Neil, 2016] O'Neil, C. (2016). *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group, USA.

[Oreopoulos, 2011] Oreopoulos, P. (2011). Why do skilled immigrants struggle in the labor market? a field experiment with thirteen thousand resumes. *American Economic Journal: Economic Policy*, 3(4):148–71.

[Pessach and Shmueli, 2022] Pessach, D. and Shmueli, E. (2022). A review on fairness in machine learning. *ACM Comput. Surv.*, 55(3).

[Potamias et al., 2009] Potamias, M., Bonchi, F., Castillo, C., and Gionis, A. (2009). Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 867–876.

[Rahman et al., 2019] Rahman, M., Hernandez, N., Recabarren, R., Ahmed, S. I., and Carbunar, B. (2019). The art and craft of fraudulent app promotion in google play. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 24372454.

[Reimers and Gurevych, 2019] Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

[Rozemberczki et al., 2021] Rozemberczki, B., Allen, C., and Sarkar, R. (2021). Multi-Scale attributed node embedding. *Journal of Complex Networks*, 9(2). cnab014.

[Schapire, 1999] Schapire, R. E. (1999). A brief introduction to boosting. IJCAI'99, page 14011406, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Schumacher et al., 2020] Schumacher, T., Wolf, H., Ritzert, M., Lemmerich, F., Bachmann, J., Frantzen, F., Klabunde, M., Grohe, M., and Strohmaier, M. (2020). The effects of randomness on the stability of node embeddings. In *PKDD/ECML Workshops*.

[Shchur et al., 2018] Shchur, O., Mumme, M., Bojchevski, A., and Gnnemann, S. (2018). Pitfalls of graph neural network evaluation.

[Shojaee et al., 2015] Shojaee, S., Azman, A., Murad, M., Sharef, N., and Sulaiman, N. (2015). A framework for fake review annotation. In *Proceedings of the 2015 17th UKSIM-AMSS International Conference on Modelling and Simulation*.

[Singh et al., 2022] Singh, A., Arun, A., Malhotra, P., Desur, P., Jain, A., Chau, D. H., and Kumaraguru, P. (2022). Install-incentivising apps on google play. `https://precog.iiit.ac.in/requester.php?dataset=google_play`.

[Spinelli et al., 2021] Spinelli, I., Scardapane, S., Hussain, A., and Uncini, A. (2021). Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning. *IEEE Transactions on Artificial Intelligence*, 3(3):344–354.

[Statista, 2022a] Statista (2022a). Global google play app downloads 2016-2021. `https://www.statista.com/statistics/734332/google-play-app-installs-per-year/`.

[Statista, 2022b] Statista (2022b). Global mobile app install advertising spending 2017-2022. `https://www.statista.com/statistics/986536/mobile-app-install-advertising-spending-global/`.

[Statista, 2022c] Statista (2022c). Google play: number of available apps 2009-2022. `https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/`.

[Suresh et al., 2021] Suresh, S., Budde, V., Neville, J., Li, P., and Ma, J. (2021). Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Data Mining*, KDD '21, page 15411551, New York, NY, USA. Association for Computing Machinery.

[Tang et al., 2020] Tang, X., Yao, H., Sun, Y., Wang, Y., Tang, J., Aggarwal, C., Mitra, P., and Wang, S. (2020). Investigating and mitigating degree-related biases in graph convoltuional networks. CIKM '20, page 14351444, New York, NY, USA. Association for Computing Machinery.

[Velickovic et al., 2018] Velickovic, P., Fedus, W., Hamilton, W. L., Li, P., Bengio, Y., and Hjelm, R. D. (2018). Deep graph infomax.

[Ying et al., 2018] Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., and Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983.

[Zafar et al., 2019] Zafar, M. B., Valera, I., Gomez-Rodriguez, M., and Gummadi, K. P. (2019). Fairness constraints: A flexible approach for fair classification. *Journal of Machine Learning Research*, 20(75):1–42.

[Zitnik and Leskovec, 2017] Zitnik, M. and Leskovec, J. (2017). Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33(14):i190–i198.