Unbiased Graph Embedding with Biased Graph Observations

Nan Wang, Lu Lin, Jundong Li, Hongning Wang
{nw6a,ll5fy,jl6qk,hw5x}@virginia.edu
University of Virginia
Charlottesville, VA, USA

ABSTRACT

Graph embedding techniques are pivotal in real-world machine learning tasks that operate on graph-structured data, such as social recommendation and protein structure modeling. Embeddings are mostly performed on the node level for learning representations of each node. Since the formation of a graph is inevitably affected by certain *sensitive node attributes*, the node embeddings can inherit such sensitive information and introduce undesirable biases in downstream tasks. Most existing works impose ad-hoc constraints on the node embeddings to restrict their distributions for unbiasedness/fairness, which however compromise the utility of the resulting embeddings. In this paper, we propose a principled new way for unbiased graph embedding by learning node embeddings from an *underlying bias-free graph*, which is not influenced by sensitive node attributes. Motivated by this new perspective, we propose two complementary methods for uncovering such an underlying graph, with the goal of introducing minimum impact on the utility of the embeddings. Both our theoretical justification and extensive experimental comparisons against state-of-the-art solutions demonstrate the effectiveness of our proposed methods.

CCS CONCEPTS

• **Computing methodologies → Machine learning**; • **Applied computing → Law, social and behavioral sciences**.

KEYWORDS

unbiased graph embedding, sensitive attributes, bias-free graph

ACM Reference Format:
Nan Wang, Lu Lin, Jundong Li, Hongning Wang. 2022. Unbiased Graph Embedding with Biased Graph Observations. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3485447.3512189

1 INTRODUCTION

Graph embedding is an indispensable building block in modern machine learning approaches that operate on graph-structured data [12, 13, 20, 35, 41]. Graph embedding methods map each node to a low-dimensional embedding vector that reflects the nodes' structural information from the observed connections in the given graph. These node embeddings are then employed to solve downstream tasks, such as friend recommendation in social networks (i.e., link prediction) or user interest prediction in e-commerce platforms (i.e., node classification) [32, 44].

However, the observed node connections in a graph are inevitably affected by certain *sensitive node attributes* (e.g., gender, age, race, religion, etc., of users) [36], which are intended to be withheld from many high-stake real-world applications. Without proper intervention, the learned node embeddings can inherit undesired sensitive information and lead to severe bias and fairness concerns in downstream tasks [5, 37]. For example, in social network recommendation, if the users with the same gender are observed to connect more often, the learned embeddings can record such information and lead to gender bias by only recommending friends to a user with the same gender identity. Biased node embeddings, when applied in applications such as loan application [22] or criminal justice [4], may unintentionally favor or disregard one demographic group, causing unfair treatments. Besides, from the data privacy perspective, this also opens up the possibility for extraction attacks from the node embeddings [39]. These realistic and ethical concerns set a higher bar for the graph embedding methods to learn both effective and unbiased embeddings.

There is rich literature in enforcing unbiasedness/fairness in algorithmic decision making, especially in classical classification problems [8, 17, 48]. Unbiased graph embedding has just started to attract research attentions in recent years. To date, the most popular recipe for unbiased graph embedding is to add adversarial regularizations to the loss function, such that the sensitive attributes cannot be predicted from the learned embeddings [1, 5, 11, 26]. For example, making a discriminator built on the node embeddings fail to predict the sensitive attributes of the nodes. However, such a regularization is only *a necessary condition* for debiasing node embeddings, and it usually hurts the utility of the embeddings (a trivial satisfying solution is to randomize the embeddings). Besides these regularization-based solutions, Fairwalk [37] modifies the random walk strategy in the node2vec algorithm [13] into two levels: when choosing the next node on a path, it first randomly selects a group defined by sensitive attributes, and then randomly samples a reachable node from that group. DeBayes [6] proposes to capture the sensitive information by a prior function in Conditional Network Embedding [18], such that the learned embeddings will not carry the sensitive information. Nevertheless, both Fairwalk and DeBayes are based on specific graph embedding methods; and how to generalize them to other types of graph embedding methods such as GAT [43] or SGC [46] is not obvious.

Moving beyond the existing unbiased graph embedding paradigm, in this paper, we propose a principled new framework for the purpose with theoretical justifications. Our solution is to learn node embeddings from an *underlying bias-free graph* whose edges

[0]Equal contribution between the first two authors.

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9096-5/22/04.
https://doi.org/10.1145/3485447.3512189

are generated without influence from sensitive attributes. Specifically, as suggested by Pfeiffer et al. [36], the generation of a graph can be treated as a two-phase procedure. In the first phase, the nodes are connected with each other solely based on global graph *structural properties*, such as degree distributions, diameter, edge connectivity, clustering coefficients and etc., resulting in an *underlying structural graph*, free of influences from node attributes. In the second phase, the connections are *re-routed* by the node attributes (including both sensitive and non-sensitive attributes). For example, in a social network, users in the same age group tend to be more connected than those in different age groups, leading to the final observed graph biased by the age attribute. Hence, our debiasing principle is to filter out the influence from sensitive attributes on the underlying structural graph to create a bias-free graph (that only has non-sensitive or no attributes) from the observed graph, and then perform embedding learning on the bias-free graph.

We propose two alternative ways to uncover the bias-free graph from the given graph for learning node embeddings. The first is a weighting-based method, which reweighs the graph reconstruction based loss function with importance sampling on each edge, such that the derived loss is as calculated on the bias-free graph, in expectation. This forms *a sufficient condition* for learning unbiased node embeddings: when the reconstruction loss is indeed defined on the corresponding bias-free graph, the resulting node embeddings are unbiased, since the bias-free graph is independent from the sensitive attributes. The second way is via regularization, in which we require that, with and without the sensitive attributes, the probabilities of generating an edge between two nodes from their embeddings are the same. In contrast, this forms *a necessary condition*: when the learning happens on the bias-free graph, the resulting embeddings should not differentiate if any sensitive attributes participated in the generation of observed graph, i.e., the predicted edge generation should be independent from the sensitive attributes. These two methods are complementary and can be combined to control the trade-off between utility and unbiasedness.

Comprehensive experiments on three datasets and several backbone graph embedding models prove the effectiveness of our proposed framework. It achieves encouraging trade-off between unbiasedness and utility of the learned embeddings. Results also suggest that the embeddings from our methods can lead to *fair* predictions in the downstream applications. In Section 2, we discuss the related work. We introduce the notation and preliminary knowledge on unbiased graph embedding in Section 3. We formally define the underlying bias-free graph in Section 4, and propose the unbiased graph embedding methods in Section 5. We evaluate the proposed methods in Section 6 and conclude in Section 7.

2 RELATED WORK

Graph embedding aims to map graph nodes to low-dimensional vector representations such that the original graph can be reconstructed from these node embeddings. Traditional approaches include matrix factorization and spectral clustering techniques [3, 31]. Recent years have witnessed numerous successful advances in deep neural architectures for learning node embeddings. Deepwalk [35] and node2vec [13] utilize a skip-gram [28] based objective to recover the node context in random walks on a graph. Graph Convolutional Networks (GCNs) learn a node's embedding by aggregating the features from its neighbors supervised by node/edge labels in an end-to-end manner. These techniques are widely applied in friend or content recommendation [25, 47], protein structure prediction [16], and many more.

Recent efforts on unbiased and fair graph embedding mainly focus on *pre-processing*, *algorithmic* and *post-processing* steps in the learning pipeline. The pre-processing solutions modify the training data to reduce the leakage of sensitive attributes [7]. Fairwalk [37] is a typical pre-processing method which modifies the sampling process of random walk on graphs by giving each group of neighboring nodes an equal chance to be chosen. However, such pre-processing may well shift the data distribution and leads the trained model to inferior accuracy and fairness measures. The *post-processing* methods employ discriminators to correct the learned embeddings to satisfy specific fairness constraints [14]. However, such ad-hoc post-correction is detached from model training which can heavily degrade model's prediction quality.

Our work falls into the category of *algorithmic* methods, which modify the learning objective to prevent bias from the node embeddings. The most popular algorithmic solution is adding (adversarial) regularizations as constraints to filter out sensitive information [1, 5, 10]. Compositional fairness constraints [5] are realized by a composition of discriminators for a set of sensitive attributes jointly trained with the graph embedding model. Similarly, FairGNN [10] adopts a fair discriminator but focuses on debiasing with missing sensitive attribute values. Different from regularization based methods. DeBayes [6] reformulates the maximum likelihood estimation with a biased prior which absorbs the information about sensitive attributes; but this solution is heavily coupled with the specific embedding method thus is hard to generalize. Our method differs from these previous works by learning embeddings from an underlying bias-free graph. We investigate the generation of the given graph and remove the influence from sensitive attributes in the generative process to uncover a bias-free graph for graph embedding.

Generative graph models [2, 36] focus on the statistical process of graph generation by modeling the joint distributions of edges conditioned on node attributes and graph structure. For instance, Attributed Graph Model (AGM) [36] jointly models graph structure and node attributes in a two step graph generation process. AGM first exploits a structural generative graph model to compute underlying edge probabilities based on the structural properties of a given graph. It then learns attribute correlations among edges from the observed graph and combines them with the structural edge probabilities to sample edges conditioned on attribute values. This process motivates us to uncover an underlying bias-free graph by separating out sensitive attributes and only conditioning on non-sensitive attributes for calculating edge probabilities.

3 PRELIMINARIES

In this section, we first introduce our notations and general graph embedding concepts. Since the bias/fairness issues emerge most notably in prediction tasks involving humans, such as loan application or job recommendation, we will use user-related graphs as running examples to discuss our criterion for unbiased graph embedding. But we have to emphasize that this setting is only to illustrate the concept of unbiased graph embedding; and our proposed solution can be applied to any graph data and selected sensitive attributes to avoid biases in the learned embeddings.

3.1 Notation

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ be an undirected, attributed graph with a set of N nodes $\mathcal{V}$, a set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and a set of N attribute vectors $\mathcal{A}$ (one attribute vector for each node). We use (u, v) to denote an edge between node u and node v. The number of attributes on each node is K, and $\mathcal{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N\}$, where $\boldsymbol{a}_u$ is a K-dimensional attribute value vector for node u. We assume all attributes are categorical and $\mathcal{S}_i$ is the set of all possible values for attribute i. [1] For example, if node u is a user node, and the i-th attribute is gender with possible values $\mathcal{S}_i = \{Female, Male, Unknown\}$, then $\boldsymbol{a}_u[i] = Female$ indicates u is a female. Without loss of generality, we assume the first m attributes are sensitive, and $a_u[: m]$ and $a_u[m :]$ stands for the m sensitive attributes and the rest of the attributes that are non-sensitive, respectively.

In the problem of graph embedding learning, we aim to learn an encoder $\text{ENC} : \mathcal{V} \to \mathbb{R}^d$ that maps each node u to a d-dimensional embedding vector $\boldsymbol{z}_u = \text{ENC}(u)$. We focus on the *unsupervised* embedding setting which does not require node labels and the embeddings are learned via the *link prediction task*. In this task, a scoring function $\text{s}_{\boldsymbol{\theta}}(\boldsymbol{z}_u, \boldsymbol{z}_v)$ with parameters $\boldsymbol{\theta}$ is defined to predict the probability of an edge $(u, v) \in \mathcal{E}$ between node u and node v in the given graph. The loss for learning node embeddings and parameters of the encoder and scoring function is defined by:

$$\sum_{(u,v) \in \mathcal{E}} \mathcal{L}_{edge}(\text{s}_{\boldsymbol{\theta}}(\boldsymbol{z}_u, \boldsymbol{z}_v)), \tag{1}$$

where $\mathcal{L}_{edge}$ is a per-edge loss function on $(u, v) \in \mathcal{E}$. Such loss functions generally aim to maximize the likelihood of observed edges in the given graph, comparing to the negative samples of node pairs where edges are not observed [13, 29].

3.2 Unbiased Graph Embedding

Given a node u, we consider its embedding $\boldsymbol{z}_u$ as unbiased with respect to an attribute i if it is independent from the attribute. Prior works evaluate such unbiasedness in the learned node embeddings by their ability to predict the values of the sensitive attributes [5, 6, 33]. For example, they first train a classifier on a subset of node embeddings using their associated sensitive attribute values as labels. If the classifier cannot correctly predict the sensitive attribute values on the rest of node embeddings, one claims that the embeddings have low bias. If the prediction performance equals to that from random node embeddings, the learned embeddings are considered bias-free. In fact, such classifiers are often used as discriminators in adversarial methods where the classifier and the embeddings are learned jointly: the embeddings are pushed in directions where the classifier has low prediction accuracy [5, 26].

There are also studies that use fairness measures such as demographic parity or equalized opportunity to define the unbiasedness of learned embeddings [6, 14]. But we need to clarify that such fairness measures can only evaluate the fairness of the final prediction results for the intended downstream tasks, but cannot assess whether the embeddings are biased by, or contain any information about, sensitive attributes. In particular, fairness in a downstream task is only a necessary condition for unbiased embedding learning, not sufficient. The logic is obvious: unbiased embeddings can lead to fair prediction results as no sensitive attribute information is involved; but obtaining fairness in one task does not suggest the embeddings themselves are unbiased, e.g., those embeddings can still lead to unfair results in other tasks or even the fair results are obtained by other means, such as post-processing of the prediction results [45]. In Section 6, we will use both the prediction accuracy on sensitive attributes and fairness measures on final tasks to evaluate the effectiveness of our unbiased graph embedding methods.

4 EFFECT OF ATTRIBUTES IN GRAPH GENERATION

In this section, we discuss the generation of an observed graph by explicitly modeling the effects of node attributes in the process. In particular, we assume that there is an *underlying structural graph* behind an observed graph, whose edge distribution is governed by the global graph structural properties such as degree distributions, diameter, and clustering coefficients. The attributes in $\mathcal{A}$ will modify the structural edge distribution based on effects like *homophily* in social networks, where links are rewired based on the attribute similarities of the individuals [23, 27]. The modified edge distribution is then used to generate the observed graph.

Formally, let $\mathcal{M}$ be a structural generative graph model and Θ_M be the set of parameters that describe properties of the underlying structural graph. In particular, this set of parameters Θ_M is independent from node attributes in $\mathcal{A}$. We consider the class of models that represent the set of possible edges in the graph as binary random variables $E_{uv}, u \in \mathcal{V}, v \in \mathcal{V}$: i.e., the event $E_{uv} = 1$ indicates $(u, v) \in \mathcal{E}$. The model $\mathcal{M}$ assigns a probability to E_{uv} based on Θ_M, $P_M(E_{uv} = 1|\Theta_M)$. Therefore, the edges of an underlying structural graph $\mathcal{G}_M$ can be considered as samples from $Bernoulli(P_M(E_{uv} = 1|\Theta_M))$. There are many such structural models $\mathcal{M}$ such as the Chung Lu model [9] and Kronecker Product Graph Model [24]. Note that $\mathcal{M}$ does not consider node attributes in the generation of the structural graph.

Now we involve the attributes in the generative process. Let $C_{uv} \in \{(\boldsymbol{a}_i, \boldsymbol{a}_j)|i \in \mathcal{V}, j \in \mathcal{V}\}$ be a random variable indicating the *attribute value combination* of a randomly sampled pair of nodes u and v, which is independent from Θ_M. Note that C_{uv} instantiated by different node pairs can be the same, as different nodes can have the same attribute values. $P_o(E_{uv} = 1|C_{uv} = \boldsymbol{a}_{uv}, \Theta_M)$ is the conditional probability of an edge (u, v) given the corresponding attribute values on the incident nodes and structural parameters Θ_M, where $\boldsymbol{a}_{uv} = (\boldsymbol{a}_u, \boldsymbol{a}_v)$ denotes the *observed* attribute value combination on nodes u and v. Based on Bayes' Theorem, we have

$$\begin{aligned} & P_o(E_{uv} = 1|C_{uv} = \boldsymbol{a}_{uv}, \Theta_M) \\ &= \frac{P_o(C_{uv} = \boldsymbol{a}_{uv}|E_{uv} = 1, \Theta_M) P_o(E_{uv} = 1|\Theta_M)}{P_o(C_{uv} = \boldsymbol{a}_{uv}|\Theta_M)} \\ &= P_M(E_{uv} = 1|\Theta_M) \frac{P_o(C_{uv} = \boldsymbol{a}_{uv}|E_{uv} = 1, \Theta_M)}{P_o(C_{uv} = \boldsymbol{a}_{uv}|\Theta_M)}, \forall u \in \mathcal{V}, \forall v \in \mathcal{V}, \end{aligned} \tag{2}$$

[1] We acknowledge that there are cases where attribute values are continuous, where discretization techniques can be applied.

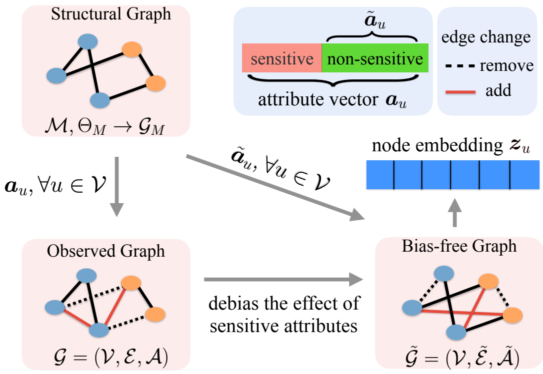


Figure 1: Illustration of Unbiased Graph Embedding (UGE). The color of the nodes represents the value of their attributes, and different line styles suggest how the observed edges are influenced by attributes in the generative process.

where the prior distribution on E_{uv} is specified by the structural model $\mathcal{M}$: i.e., $P_o(E_{uv} = 1|\Theta_M) = P_M(E_{uv} = 1|\Theta_M)$, and the posterior distribution accounts for the influences from the attribute value combinations. Therefore, the edge probabilities used to generate the observed graph with node attributes is a *modification* of those from a structural graph defined by $\mathcal{M}$ and Θ_M. It is important to clarify that the node attributes are given ahead of graph generation. They are the input to the generative process, not the output. Hence, $P_o(C_{uv} = \boldsymbol{a}_{uv}|E_{uv} = 1, \Theta_M)$ represents the probability that in all edges, a specific attribute value combination $\boldsymbol{a}_{uv}$ is observed on an edge's incident nodes. It is thus the same for all edges whose incident nodes have the same attribute value combination.

To simplify the notation, let us define a function that maps the attribute value combination $\boldsymbol{a}_{uv}$ to the probability ratio that modifies the structural graph into the observed graph by

$$R(\boldsymbol{a}_{uv}) := \frac{P_o(C_{uv} = \boldsymbol{a}_{uv}|E_{uv} = 1, \Theta_M)}{P_o(C_{uv} = \boldsymbol{a}_{uv}|\Theta_M)}, \forall u \in \mathcal{V}, \forall v \in \mathcal{V}.$$

Thus we can rewrite Eq (2) by

$$P_o(E_{uv} = 1|C_{uv} = \boldsymbol{a}_{uv}, \Theta_M) = P_M(E_{uv} = 1|\Theta_M) R(\boldsymbol{a}_{uv}). \quad (3)$$

In this way, we explicitly model the effect of node attributes by $R(\boldsymbol{a}_{uv})$, which modifies the structural graph distribution $P_M(E_{uv} = 1|\Theta_M)$ for generating the observed graph $\mathcal{G}$.

5 UNBIASED GRAPH EMBEDDING FROM A BIAS-FREE GRAPH

In this section, we describe our proposed methods for learning unbiased node embeddings based on the generative modeling of the effects of sensitive attributes in Section 4. In a nutshell, we aim to get rid of the sensitive attributes and modify the structural edge probabilities by only conditioning on non-sensitive attributes. This gives us the edge probabilities of a bias-free graph, from which we can learn unbiased node embeddings. We illustrate this principle in Figure 1. Consider a world without the sensitive attributes, and the attribute vector of node u becomes $\tilde{\boldsymbol{a}}_u = \boldsymbol{a}_u[m:]$, which only include non-sensitive attributes in $\boldsymbol{a}_u$. We denote $\tilde{\mathcal{G}} = (\mathcal{V}, \tilde{\mathcal{E}}, \tilde{\mathcal{A}})$ as the corresponding new graph generated with $\tilde{\boldsymbol{a}}_u, \forall u \in \mathcal{V}$, and $\tilde{\boldsymbol{a}}_{uv} = (\tilde{\boldsymbol{a}}_u, \tilde{\boldsymbol{a}}_v)$. Therefore, $\tilde{\mathcal{G}}$ is a bias-free graph without influence from sensitive attributes. If we can learn node embeddings from $\tilde{\mathcal{G}}$ instead of $\mathcal{G}$, the embeddings are guaranteed to be unbiased with respect to sensitive attributes. Specifically, the edge probabilities used for generating $\tilde{\mathcal{G}}$ can be written as

$$P_{\tilde{o}}(E_{uv} = 1|\tilde{C}_{uv} = \tilde{\boldsymbol{a}}_{uv}, \Theta_M) = P_M(E_{uv} = 1|\Theta_M)\tilde{R}(\tilde{\boldsymbol{a}}_{uv}), \quad (4)$$

where

$$\tilde{R}(\tilde{\boldsymbol{a}}_{uv}) := \frac{P_{\tilde{o}}(\tilde{C}_{uv} = \tilde{\boldsymbol{a}}_{uv}|E_{uv} = 1, \Theta_M)}{P_{\tilde{o}}(\tilde{C}_{uv} = \tilde{\boldsymbol{a}}_{uv}|\Theta_M)}, \forall u \in \mathcal{V}, \forall v \in \mathcal{V}, \quad (5)$$

$\tilde{C}_{uv} \in \{(\tilde{\boldsymbol{a}}_i, \tilde{\boldsymbol{a}}_j)|i \in \mathcal{V}, j \in \mathcal{V}\}$ is the random variable indicating attribute value combinations without sensitive attributes, and $P_{\tilde{o}}$ indicates the distributions used in generating $\tilde{\mathcal{G}}$. We name the class of methods that learn embeddings from $\tilde{\mathcal{G}}$ as UGE, simply for **U**nbiased **G**raph **E**mbedding. Next we introduce two instances of UGE. The first is UGE-W, which reweighs the per-edge loss such that the total loss is from $\tilde{\mathcal{G}}$ in expectation. The second method is UGE-R, which adds a regularization term to shape the embeddings to satisfy the properties as those directly learned from $\tilde{\mathcal{G}}$.

5.1 Weighting-Based UGE

To compose a loss based on $\tilde{\mathcal{G}}$, we modify the loss function in Eq (1) by reweighing the loss term on each edge as

$$\mathcal{L}_{UGE-W}(\mathcal{G}) = \sum_{(u,v)\in\mathcal{E}} \mathcal{L}_{edge}(s_{\boldsymbol{\theta}}(\boldsymbol{z}_u, \boldsymbol{z}_v)) \frac{\tilde{R}(\tilde{\boldsymbol{a}}_{uv})}{R(\boldsymbol{a}_{uv})}. \quad (6)$$

The following theorem shows that, in expectation, this new loss is equivalent to the loss for learning node embeddings from $\tilde{\mathcal{G}}$.

Theorem 5.1. *Given a graph $\mathcal{G}$, and $\tilde{R}(\tilde{\boldsymbol{a}}_{uv})/R(\boldsymbol{a}_{uv}), \forall(u,v) \in \mathcal{E}$, $\mathcal{L}_{UGE-W}(\mathcal{G})$ is an unbiased loss with respect to $\tilde{\mathcal{G}}$.*

Proof. We take expectation over the edge observations in $\mathcal{G}$ as

$$\begin{aligned} & \mathbb{E}\left[\mathcal{L}_{UGE-W}(\mathcal{G})\right] \quad (7) \\ =& \mathbb{E}\left[\sum_{(u,v)\in\mathcal{E}} \mathcal{L}_{edge}(s(\boldsymbol{z}_u, \boldsymbol{z}_v)) \frac{\tilde{R}(\tilde{\boldsymbol{a}}_{uv})}{R(\boldsymbol{a}_{uv})}\right] \\ =& \mathbb{E}\left[\sum_{u\in\mathcal{V}, v\in\mathcal{V}} \mathcal{L}_{edge}(s(\boldsymbol{z}_u, \boldsymbol{z}_v)) \frac{\tilde{R}(\tilde{\boldsymbol{a}}_{uv})}{R(\boldsymbol{a}_{uv})} \cdot E_{uv}\right] \\ =& \sum_{u\in\mathcal{V}, v\in\mathcal{V}} \mathcal{L}_{edge}(s(\boldsymbol{z}_u, \boldsymbol{z}_v)) \frac{\tilde{R}(\tilde{\boldsymbol{a}}_{uv})}{R(\boldsymbol{a}_{uv})} \cdot P_o(E_{uv} = 1|C_{uv} = \boldsymbol{a}_{uv}, \Theta_M) \\ * =& \sum_{u\in\mathcal{V}, v\in\mathcal{V}} \mathcal{L}_{edge}(s(\boldsymbol{z}_u, \boldsymbol{z}_v)) \cdot P_{\tilde{o}}(E_{uv} = 1|\tilde{C}_{uv} = \tilde{\boldsymbol{a}}_{uv}, \Theta_M) \\ =& \mathbb{E}\left[\sum_{(u,v)\in\tilde{\mathcal{E}}} \mathcal{L}_{edge}(s(\boldsymbol{z}_u, \boldsymbol{z}_v))\right]. \end{aligned}$$

The step marked by $*$ uses Eq (3) and Eq (4). □

UGE-W is closely related to the idea of importance sampling [21], which analyzes the edge distribution of the bias-free graph $\tilde{\mathcal{G}}$ by observations from the given graph $\mathcal{G}$. The only thing needed for deploying UGE-W in existing graph embedding methods is to calculate the weights $\tilde{R}(\tilde{\boldsymbol{a}}_{uv})/R(\boldsymbol{a}_{uv})$. To estimate $R(\boldsymbol{a}_{uv})$, we need the estimates of $P_o(C_{uv} = \boldsymbol{a}_{uv}|E_{uv} = 1, \Theta_M)$ and $P_o(C_{uv} =$

$\boldsymbol{a}_{uv}|\Theta_M)$. With maximum likelihood estimates on the observed graph, we have

$$P_o(C_{uv}=\boldsymbol{a}_{uv}|E_{uv}=1,\Theta_M)\approx\frac{\sum_{(i,j)\in\mathcal{E}}\mathbb{I}[\boldsymbol{a}_{ij}=\boldsymbol{a}_{uv}]}{|\mathcal{E}|},\quad(8)$$

$$P_o(C_{uv}=\boldsymbol{a}_{uv}|\Theta_M)\approx\frac{\sum_{i\in\mathcal{V},j\in\mathcal{V}}\mathbb{I}[\boldsymbol{a}_{ij}=\boldsymbol{a}_{uv}]}{N^2}.\quad(9)$$

Similarly we can estimate $\tilde{R}(\tilde{\boldsymbol{a}}_{uv})$ by

$$P_{\tilde{o}}(\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv}|E_{uv}=1,\Theta_M)\approx\frac{\sum_{(i,j)\in\tilde{\mathcal{E}}}\mathbb{I}[\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}]}{|\tilde{\mathcal{E}}|},\quad(10)$$

$$P_{\tilde{o}}(\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv}|\Theta_M)\approx\frac{\sum_{i\in\mathcal{V},j\in\mathcal{V}}\mathbb{I}[\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}]}{N^2}.\quad(11)$$

Note that the estimation of $P_{\tilde{o}}(\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv}|E_{uv}=1,\Theta_M)$ is based on $\tilde{\mathcal{E}}$, which is unfortunately from the implicit bias-free graph $\tilde{\mathcal{G}}$ and unobservable. But we can approximate it with $\mathcal{E}$ in the following way: after grouping node pairs by non-sensitive attribute value combinations $\tilde{\boldsymbol{a}}_{uv}$, the sensitive attributes only re-route the edges but do not change the number of edges in each group. Thus,

$$\begin{aligned}P_{\tilde{o}}(\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv}|E_{uv}=1,\Theta_M)&\approx\frac{\sum_{(i,j)\in\tilde{\mathcal{E}}}\mathbb{I}[\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}]}{|\tilde{\mathcal{E}}|}\quad(12)\\&=\frac{\sum_{i\in\mathcal{V},j\in\mathcal{V},\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}}\mathbb{I}[(i,j)\in\tilde{\mathcal{E}}]}{|\tilde{\mathcal{E}}|}\\&=\frac{\sum_{i\in\mathcal{V},j\in\mathcal{V},\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}}\mathbb{I}[(i,j)\in\mathcal{E}]}{|\tilde{\mathcal{E}}|}\\&=\frac{\sum_{(i,j)\in\mathcal{E}}\mathbb{I}[\tilde{\boldsymbol{a}}_{ij}=\tilde{\boldsymbol{a}}_{uv}]}{|\mathcal{E}|}.\end{aligned}$$

For node pairs with the same attribute value combination, Eq (8)-Eq (11) only need to be calculated once instead of for each pair. This can be done by first grouping node pairs by their attribute value combinations and then perform estimation in each group. However, when there are many attributes or attributes can take many unique values, the estimates may become inaccurate since there will be many groups and each group may only have a few nodes. In this case, we can make independence assumptions among the attributes. For example, by assuming they are independent, the estimate for a specific attribute value combination over all the K attributes becomes the product of K estimates, one for each attribute. The non-sensitive attributes can be safely removed under this assumption with $\tilde{R}(\tilde{\boldsymbol{a}}_{uv})=1$, and only $R(\boldsymbol{a}_{uv})$ needs to be estimated as $R(\boldsymbol{a}_{uv})=\prod_{i=1}^{m}R(\boldsymbol{a}_{uv}[i])$. Since UGE-W only assigns pre-computed weights to the loss, the optimization based on it will not increase the complexity of any graph embedding method.

5.2 Regularization-Based UGE

We propose an alternative way for UGE which adds a regularization term to the loss function that pushes the embeddings to satisfy properties required by the bias-free graph $\tilde{\mathcal{G}}$. Specifically, when the node embeddings are learned from $\tilde{\mathcal{G}}$, their produced edge distributions should be the same with and without the sensitive attributes. To enforce this condition, we need to regularize the discrepancy between $P_o(E_{uv}=1|C_{uv}=\boldsymbol{a}_{uv},\Theta_M)$ and $P_{\tilde{o}}(E_{uv}=1|\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv},\Theta_M)$ induced from the node embeddings. We can use the scores in $s_{\theta}(z_u,z_v)$ as a proxy to represent edge probability produced by the embeddings of nodes u and v, i.e., high $s_{\theta}(z_u,z_v)$ indicates high probability of an edge between u and v. We can measure $P_o(E_{uv}=1|C_{uv}=\boldsymbol{a}_{uv},\Theta_M)$ by aggregating node pairs with the same attribute value combination to marginalize out the effect of Θ_M and focus on the influence from attributes as

$$Q_{\boldsymbol{a}_{uv}}=\frac{1}{N_{\boldsymbol{a}_{uv}}}\sum_{i\in\mathcal{V},j\in\mathcal{V},\boldsymbol{a}_{ij}=\boldsymbol{a}_{uv}}s_{\theta}(z_i,z_j),\quad(13)$$

where we use $Q_{\boldsymbol{a}_{uv}}$ to denote the approximated measure of $P_o(E_{uv}=1|C_{uv}=\boldsymbol{a}_{uv},\Theta_M)$, and $N_{\boldsymbol{a}_{uv}}$ is the number of node pairs that has the attribute value combination $\boldsymbol{a}_{uv}$. For pairs with the same attribute value combination, $Q_{\boldsymbol{a}_{uv}}$ only needs to be calculated once. Similarly, $P_{\tilde{o}}(E_{uv}=1|\tilde{C}_{uv}=\tilde{\boldsymbol{a}}_{uv},\Theta_M)$ can be represented by $Q_{\tilde{\boldsymbol{a}}_{uv}}$, which can be obtained by aggregating the scores over pairs with non-sensitive attribute value combination $\tilde{\boldsymbol{a}}_{uv}$. Finally, we use ℓ_2 distance between $Q_{\boldsymbol{a}_{uv}}$ and $Q_{\tilde{\boldsymbol{a}}_{uv}}$ as the regularization

$$\begin{aligned}&\mathcal{L}_{UGE-R}(\mathcal{G})\quad(14)\\&=\sum_{(u,v)\in\mathcal{E}}\mathcal{L}_{edge}(s_{\theta}(z_u,z_v))+\lambda\sum_{u\in\mathcal{V},v\in\mathcal{V}}\left\|Q_{\boldsymbol{a}_{uv}}-Q_{\tilde{\boldsymbol{a}}_{uv}}\right\|_2,\end{aligned}$$

where λ controls the trade-off between the per-edge losses and the regularization.

In contrast to adversarial regularizations employed in prior work [1, 5, 11, 26], UGE-R takes a different perspective in regularizing the discrepancy between graphs with and without sensitive attributes induced from the embeddings. All previous regularization-based methods impose the constraint on individual edges. We should note that the regularization term is summed over all node pairs, which has a complexity of $O(N^3)$ and can be costly to calculate. But in practice, we can add the regulariztaion by only sampling batches of node pairs in each iteration during model update, and use λ to compensate the strength of the regularization.

5.3 Combined Method

As hinted in section 1, UGE-W is a sufficient condition for unbiased graph embedding, since it directly learns node embeddings from a bias-free graph. UGE-R is a necessary condition, as it requires the learned embeddings to satisfy the properties of a bias-free graph. We can combine them to trade-off the debiasing effect and utility,

$$\begin{aligned}&\mathcal{L}_{UGE-C}(\mathcal{G})\quad(15)\\&=\sum_{(u,v)\in\mathcal{E}}\mathcal{L}_{edge}(s_{\theta}(z_u,z_v))\frac{\tilde{R}(\tilde{\boldsymbol{a}}_{uv})}{R(\boldsymbol{a}_{uv})}+\lambda\sum_{u\in\mathcal{V},v\in\mathcal{V}}\left\|Q_{\boldsymbol{a}_{uv}}-Q_{\tilde{\boldsymbol{a}}_{uv}}\right\|_2,\end{aligned}$$

where we use $\mathcal{L}_{UGE-C}(\mathcal{G})$ to represent the combined method. $\mathcal{L}_{UGE-C}(\mathcal{G})$ thus can leverage the advantages of both UGE-W and UGE-R to achieve better trade-offs between the unbiasedness and the utility of node embeddings in downstream tasks.

6 EXPERIMENTS

In this section, we study the empirical performance of UGE on three benchmark datasets in comparison to several baselines. In particular, we apply UGE to *five* popularly adopted backbone graph embedding models to show its wide applicability. To evaluate the debiasing performance, the node embeddings are firstly evaluated

Table 1: Statistics of evaluation graph datasets.

Statistics	Pokec-z	Pokec-n	MovieLens-1M
# of nodes	67,796	66,569	9,992
# of edges	882,765	729,129	1,000,209
Density	0.00019	0.00016	0.01002

by their ability to predict the value of sensitive attributes, where lower prediction performance means better debiasing effect. Then a task-specific metric is used to evaluate the utility of the embeddings. Besides, we also apply fairness metrics in the link prediction results to demonstrate the potential of using embeddings from UGE to achieve fairness in downstream tasks.

6.1 Setup

• **Dataset.** We use three public user-related graph datasets, Pokec-z, Pokec-n and MovieLens-1M, where the users are associated with sensitive attributes to be debiased. The statistics of these three datasets are summarized in Table 1. Pokec[2] is an online social network in Slovakia, which contains anonymized data of millions of users [40]. Based on the provinces where users belong to, we used two sampled datasets named as **Pokec-z** and **Pokec-n** adopted from [10], which consist of users belonging to two major regions of the corresponding provinces, respectively. In both datasets, each user has a rich set of features, such as education, working field, interest, etc.; and we include *gender, region* and *age* as (sensitive) attributes whose effect will be studied in our evaluation. **MovieLens-1M**[3] is a popular movie recommendation benchmark, which contains around one million user ratings on movies [15]. In our experiment, we construct a bipartite graph which consists of user and movie nodes and rating relations as edges. The dataset includes *gender, occupation* and *age* information about users, which we treat as sensitive attributes to be studied. We do not consider movie attributes, and thus when applying UGE, only user attributes are counted for our debiasing purpose.

• **Graph embedding models.** UGE is a general recipe for learning unbiased node embeddings, and can be applied to different graph embedding models. We evaluate its effectiveness on five representative embedding models in the supervised setting with the link prediction task. **GCN** [19], **GAT** [42], **SGC** [46] and **node2vec** [13] are deep learning models, and we use dot product between two node embeddings to predict edge probability between them and apply cross-entropy loss for training. **MF** [30] applies matrix factorization to the adjacency matrix. Each node is represented by an embedding vector learned with pairwise logistic loss [38].

• **Baselines.** We consider three baselines for generating unbiased node embeddings. (1) **Fairwalk** [37] is based on node2vec, which modifies the pre-processing of random-walk generation by grouping neighboring nodes with their values of the sensitive attributes. Instead of randomly jumping to a neighbor node, Fairwalk firstly jumps to a group and then sample a node from that group for generating random walks. We extend it to GCN, GAT and SGC by sampling random walks of size 1 to construct the corresponding per-edge losses for these embedding models. (2) **Compositional Fairness Constraints (CFC)** [5] is an algorithmic method, which adds an adversarial regularizer to the loss by jointly training a composition of sensitive attribute discriminators. We apply CFC to all graph embedding models and tune the weight on the regularizer, where larger weights are expected to result in embeddings with less bias but lower utility. (3) **Random** embeddings are considered as a bias-free baseline. We generate random embeddings by uniformly sampling the value of each embedding dimension from [0, 1].

It is worth mentioning that a recent work **DeBayes** [6], which is based on the conditional network embedding (CNE) [18], includes the sensitive information in a biased prior for learning unbiased node embeddings. We did not include it since it is limited to CNE and cannot be easily generalized to other graph embedding models. Besides, we found the bias prior calculation in DeBayes does not scale to large graphs where the utility of resulting node embeddings is close to random. The original paper [6] only experimented with two small graph datasets with less than $4K$ nodes and $100K$ edges. By default, UGE follows Fairwalk to debias each of the sensitive attributes separately in experiments without independence assumption between attributes. CFC debiases all sensitive attributes jointly as suggested in the original paper.[4]

• **Configurations.** For the Pokec-z and Pokec-n datasets, we apply GCN, GAT, SGC and node2vec as embedding models and apply debiasing methods on top of them. For each dataset, we construct positive examples for each node by collecting N_{pos} neighboring nodes with N_{pos} equal to its node degree, and randomly sample $N_{neg} = 20 \times N_{pos}$ unconnected nodes as negative examples. For each node, we use 90% positive and negative examples for training and reserve the rest 10% for testing. For Movielens-1M, we follow common practices and use MF as the embedding model [5, 37]. We do not evalaute Fairwalk on this dataset since there is no user-user connections and fair random walk cannot be directly applied. The rating matrix is binarized to create a bipartite user-movie graph for MF. We use 80% ratings for training and 20% for testing. For all datasets and embedding models, we set the node embedding size to $d = 16$. We include more details about model implementations and hyper-parameter tuning in Appendix A.

In Section 6.2, we compare the unbiasedness and utility of embeddings from different baselines. We evaluate fairness resulted from the embeddings in Section 6.3. We study the unbiasedness-utility trade-off in UGE and CFC in Section 6.4. Since there is a large number of experimental settings composed of different datasets, embedding models, and baselines, we report results from different combinations in each section to maximize the coverage in each component, and include the other results in Appendix B.

6.2 Unbiasedness and Utility Trade-off

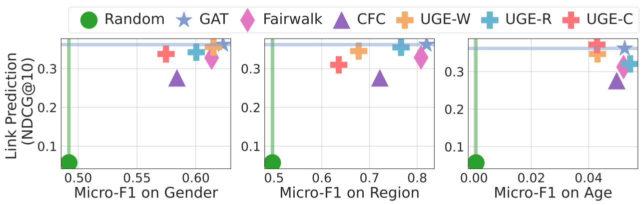
We firstly compare the unbiasedness of node embeddings from different debiasing methods. For each sensitive attribute, we train a logistic classifier with 80% of the nodes using their embeddings as features and attribute values as labels. We then use the classifier to predict the attribute values on the rest of 20% nodes and evaluate the performance with Micro-F1. The Micro-F1 score can be used to measure the severity of bias in the embeddings, i.e., a lower

[2] https://snap.stanford.edu/data/soc-pokec.html
[3] https://grouplens.org/datasets/movielens/1m/

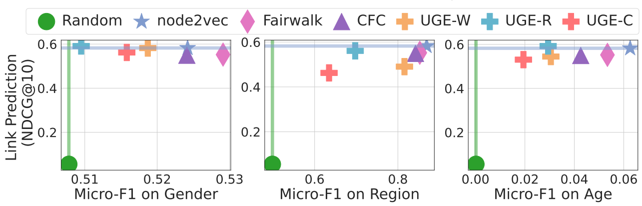
[4] UGE can debias either a single attribute or multiple attributes jointly by removing one or more attributes in the bias-free graph.

Table 2: Unbiasedness evaluated by Micro-F1 on Pokec-z and Pokec-n. Bold numbers highlight the best in each row.

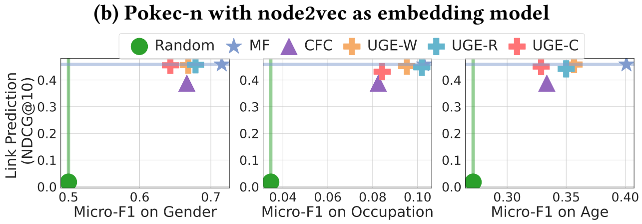
Dataset	Embedding Model	Prediction Target	No Debiasing	Fairwalk	CFC	UGE-W	UGE-R	UGE-C	Random
Pokec-z	GAT	Gender (Micro-F1)	0.6232	0.6135	0.5840	0.6150	0.6094	**0.5747**	0.4921
		Region (Micro-F1)	0.8197	0.8080	0.7217	0.6784	0.7660	**0.6356**	0.4966
		Age (Micro-F1)	0.0526	0.0522	0.0498	0.0431	0.0545	**0.0429**	0.0007
Pokec-n	node2vec	Gender (Micro-F1)	0.5241	0.5291	0.5241	0.5187	**0.5095**	0.5158	0.5078
		Region (Micro-F1)	0.8690	0.8526	0.8423	0.8158	0.6975	**0.6347**	0.4987
		Age (Micro-F1)	0.0626	0.0534	0.0426	0.0305	0.0294	**0.0194**	0.0002



(a) Pokec-z with GAT as embedding model



(b) Pokec-n with node2vec as embedding model



(c) MovieLens-1M with MF as embedding model

Figure 2: Trade-off between the utility (by NDCG@10) and unbiasedness (by Micro-F1) of different methods. Random embeddings give the lowest Micro-F1 (green line), and no debiasing gives the best NDCG@10 (blue line). An ideal debiasing method should locate itself at the upper left corner.

score means lower bias in the embeddings. Random embeddings are expected to have the lowest Micro-F1 and embeddings without debiasing should have the highest Micro-F1. We show the results on Pokec-z with GAT as base embedding model and Pokec-n with node2vec as the base embedding model in Table 2. From the results, we see that embeddings from UGE methods always have the least bias against all baselines with respect to all sensitive attributes and datasets. This confirms the validity of learning unbiased embeddings from a bias-free graph. Besides, by combining UGE-W and UGE-R, UGE-C usually produces the best debiasing effect, which demonstrates the complementary effect of the two methods.

Besides the unbiasedness, the learned embeddings need to be effective when applied to downstream tasks. In particular, we use NDCG@10 evaluated on the link prediction task to measure the

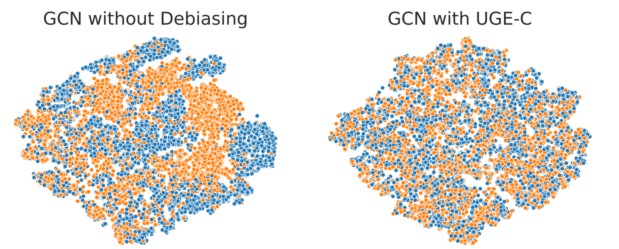


Figure 3: Visualization of embeddings learned on Pokec-n. Node color represents the region of the nodes.

utility of the embeddings. Specifically, for each target node, we create a candidate list of 100 nodes that includes all its observed neighbor nodes in the test set and randomly sampled negative nodes. Then NDCG@10 is evaluated on this list with predicted edge probabilities from the node embeddings. Figures 2a and 2b show the unbiasedness as well as the utility of embeddings from different methods in correspondence to the two datasets and embedding models in Table 2. Figure 2c shows the results on MovieLens-1M with MF as the embedding model.

In these plots, different embedding methods are represented by different shapes in the figures, and we use different colors to differentiate UGE-W, UGE-R and UGE-C. Random embeddings do not have any bias and provide the lowest Micro-F1 (green line), while embeddings without any debiasing gives the highest NDCG@10 (blue line). To achieve the best utility-unbiasedness trade-off, an ideal debiasing method should locate itself at the upper left corner. As shown in the figures, UGE based methods achieve the most encouraging trade-offs on these two contradicting objectives in most cases. UGE-C can usually achieve better debiasing effect, without sacrificing too much utility. UGE-W and UGE-R maintain high utility but are less effective than the combined version. CFC can achieve descent unbiasedness in embeddings, but the utility is seriously compromised (such as in Pokec-z and MovieLens-1M). Fairwalk unfortunately does not present an obvious debiasing effect.

To further visualize the debiasing effect of UGE, we use t-SNE to project the node embeddings on Pokec-n to a 2-D space in Figure 3. The left plot shows the embeddings learned via GCN without debiasing, and the right plot exhibits the debiased embeddings by applying UGE-C on GCN to debias the region attibute. Node colors represent the region value. Without debiasing, the embeddings are clearly clustered to reflect the regions of nodes. With UGE-C, embeddings from different regions are blended together, showing the effect of removing the region information from the embeddings.

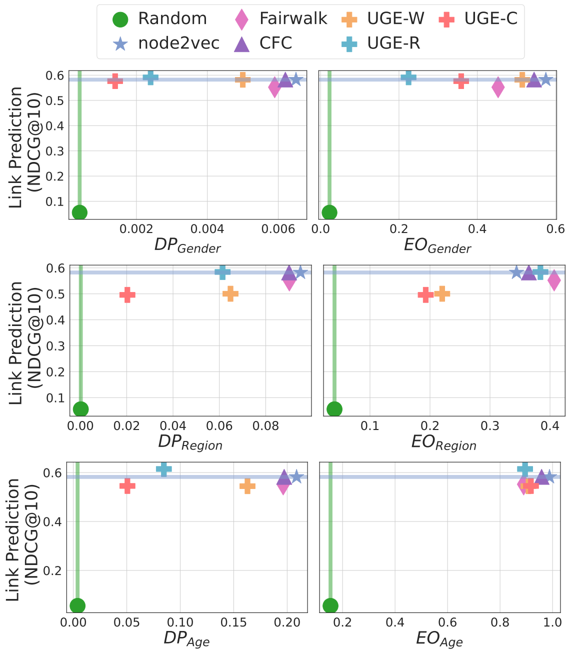


Figure 4: Fairness metrics evaluated on link prediction task on Pokec-n with node2vec as the embedding model.

6.3 High-Level Fairness from Embeddings

We study whether the debiased embeddings can lead to fairness in downstream tasks. We adopt two popular metrics—*demographic parity* (DP) and *equalized opportunity* (EO) to evaluate the fairness of link prediction results from the embeddings. DP requires that the predictions are independent from sensitive attributes, measured by the maximum difference of prediction rates between different combinations of sensitive attribute values. EO measures the independence between true positive rate (TPR) of predicted edges and sensitive attributes. It is defined by the maximum difference of TPRs between different sensitive attribute value combinations. For both DP and EO, lower values suggest better fairness. We use the exact formulation of DP and EO in [6] and use the sigmoid function to convert the edge score for a pair of nodes to a probability.

We show the results on fairness vs., utility in Figure 4, which are evaluated on each of the three sensitive attributes in Pokec-n with node2vec as the embedding model. In each plot, x-axis is the DP or EO and y-axis is the NDCG@10 on link prediction. Similar to Figure 2, the ideal debiasing methods should locate at the upper left corner. Except for EO on the *age* attribute where all methods performs similarly, UGE methods can achieve significantly better fairness than the baselines on both DP and EO, while maintaining competitive performance on link prediction. UGE-C can achieve the most fair predictions. This study shows UGE's ability to achieve fairness in downstream tasks by effectively eliminating bias in the learned node embeddings.

6.4 Unbiasedness-Utility Tradeoff in UGE

Last but not least, we study the unbiasedness-utility trade-off in UGE-C by tuning the weight on regularization. Although UGE-W

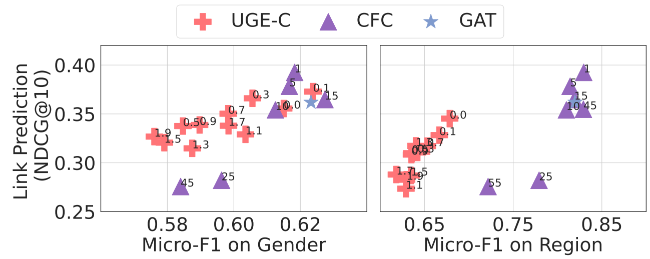


Figure 5: Trade-off comparison between CFC and UGE-C on Pokec-z with GAT as the embedding model.

itself can already achieve promising debiasing effect, we expect that the added regularization from UGE-R can complement it for a better trade-off. In particular, we tune the regularization weights in both CFC and UGE-C and plot Micro-F1 (x-axis) vs. NDCG@10 (y-axis) from the resulting embeddings in Figure 5. Weight values are marked on each point and also listed in Appendix A. The results are obtained on Pokec-z with GAT as the embedding model and the two figures correspond to debiasing *gender* and *region*, respectively. With the same extent of bias measured by Micro-F1, embeddings from UGE-C have a much higher utility as indicated by the vertical grids. On the other hand, embeddings from UGE-C have much less bias when the utility is the same as CFC, as indicated by horizontal grids. This experiment proves a better trade-off achieved in UGE-C, which is consistent with our designs on UGE-W and UGE-R. UGE-W learns from a bias-free graph without any constraints, and it is *sufficient* to achieve unbiasedness without hurting the utility of the embeddings. UGE-R constrains the embeddings to have the properties of those learned from a bias-free graph, which is *necessary* for the embeddings to be unbiased.

7 CONCLUSION

We propose a principled new way for learning unbiased node embeddings from graphs biased by sensitive attributes. The idea is to infer a bias-free graph where the influence from sensitive attributes is removed, and then learn the node embeddings from it. This new perspective motivates our design of UGE-W, UGE-R and their combined methods UGE-C. Extensive experiment results demonstrated strong debiasing effect from UGE as well as better unbiasedness-utility trade-offs in downstream applications.

We expect the principle of UGE can inspire better future designs for learning unbiased node embeddings from bias-free graphs. For example, instead of modeling the generation process and perform debiasing statistically, we can directly generate one or multiple bias-free graphs from the underlying generative graph model, and perform graph embedding on them. The regularization UGE-R can be refined with better moment matching mechanism than minimizing the l_2 distance. The weights in UGE-W can be modeled and learned for better debiasing effects. Besides, it is possible and promising to directly design unbiased GNN models that directly aggregate edges based on the inferred bias-free graph.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under grant IIS-1553568, IIS-1718216, IIS-2007492, and IIS-2006844.

REFERENCES

[1] Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. 2021. Towards a Unified Framework for Fair and Stable Graph Representation Learning. In *Proceedings of Conference on Uncertainty in Artificial Intelligence, UAI*.

[2] Edoardo Maria Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. 2008. Mixed membership stochastic blockmodels. *Journal of machine learning research* (2008).

[3] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering.. In *Nips*, Vol. 14. 585–591.

[4] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. 2021. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research* 50, 1 (2021), 3–44.

[5] Avishek Joey Bose and William L. Hamilton. 2019. Compositional Fairness Constraints for Graph Embeddings. arXiv:1905.10674 [cs.LG]

[6] Maarten Buyl and Tijl De Bie. 2020. DeBayes: a Bayesian Method for Debiasing Network Embeddings. In *International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*. 1220–1229.

[7] Flavio P Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. 2017. Optimized pre-processing for discrimination prevention. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 3995–4004.

[8] Alexandra Chouldechova. 2016. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. arXiv:1610.07524 [stat.AP]

[9] Fan Chung and Linyuan Lu. 2002. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences* (2002).

[10] Enyan Dai and Suhang Wang. 2020. Learning Fair Graph Neural Networks with Limited and Private Sensitive Attribute Information. *arXiv preprint arXiv:2009.01454* (2020).

[11] Enyan Dai and Suhang Wang. 2021. Say No to the Discrimination: Learning Fair Graph Neural Networks with Limited Sensitive Attribute Information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 680–688.

[12] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (Jul 2018), 78–94.

[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[14] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems* 29 (2016), 3315–3323.

[15] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[16] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.

[17] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2012. Fairness-Aware Classifier with Prejudice Remover Regularizer. In *Machine Learning and Knowledge Discovery in Databases*, Peter A. Flach, Tijl De Bie, and Nello Cristianini (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 35–50.

[18] Bo Kang, Jefrey Lijffijt, and Tijl De Bie. 2018. Conditional Network Embeddings. arXiv:1805.07544 [stat.ML]

[19] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[20] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907 [cs.LG]

[21] T. Kloek and H. K. van Dijk. 1978. Bayesian Estimates of Equation System Parameters: An Application of Integration by Monte Carlo. *Econometrica* 46, 1 (1978), 1–19.

[22] Carol T Kulik, L Robert Jr, et al. 2000. Demographics in service encounters: effects of racial and gender congruence on perceived fairness. *Social Justice Research* 13, 4 (2000), 375–402.

[23] Timothy La Fond and Jennifer Neville. 2010. Randomization Tests for Distinguishing Social Influence and Homophily Effects. In *Proceedings of the 19th International Conference on World Wide Web* (Raleigh, North Carolina, USA) *(WWW '10)*. Association for Computing Machinery, New York, NY, USA, 601–610.

[24] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker Graphs: An Approach to Modeling Networks. *J. Mach. Learn. Res.* 11 (March 2010), 985–1042.

[25] Peng Liu, Lemei Zhang, and Jon Atle Gulla. 2019. Real-time social recommendation based on graph embedding and temporal context. *International Journal of Human-Computer Studies* 121 (2019), 58–72.

[26] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. 2018. Learning Adversarially Fair and Transferable Representations. arXiv:1802.06309 [cs.LG]

[27] Miller McPherson, Lynn Smith-Lovin, and James M. Cook. 2001. Birds of a Feather: Homophily in Social Networks. *Review of Sociology* 27 (2001), 415–444.

[28] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[29] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc.

[30] Andriy Mnih and Russ R Salakhutdinov. 2008. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis (Eds.), Vol. 20. Curran Associates, Inc.

[31] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*. 849–856.

[32] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1105–1114.

[33] John Palowitch and Bryan Perozzi. 2020. MONET: Debiasing Graph Embeddings via the Metadata-Orthogonal Training Unit. arXiv:1909.11793 [cs.LG]

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[35] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 701–710.

[36] Joseph J. Pfeiffer, Sebastian Moreno, Timothy La Fond, Jennifer Neville, and Brian Gallagher. 2014. Attributed Graph Models: Modeling Network Structure with Correlated Attributes. In *Proceedings of the 23rd International Conference on World Wide Web* (Seoul, Korea) *(WWW '14)*. Association for Computing Machinery, New York, NY, USA, 831–842.

[37] Tahleen Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. 2019. Fairwalk: Towards Fair Graph Embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 3289–3295.

[38] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. arXiv:1205.2618 [cs.IR]

[39] Lichao Sun, Yingtong Dou, Carl Yang, Ji Wang, Philip S Yu, Lifang He, and Bo Li. 2018. Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528* (2018).

[40] Lubos Takac and Michal Zabovsky. 2012. Data analysis in public social networks. In *International scientific conference and international workshop present day trends of innovations*.

[41] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-Scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web* (Florence, Italy) *(WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1067–1077.

[42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. arXiv:1710.10903 [stat.ML]

[44] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1225–1234.

[45] Blake Woodworth, Suriya Gunasekar, Mesrob I. Ohannessian, and Nathan Srebro. 2017. Learning Non-Discriminatory Predictors. In *Proceedings of the 2017 Conference on Learning Theory (Proceedings of Machine Learning Research, Vol. 65)*, Satyen Kale and Ohad Shamir (Eds.). PMLR, 1920–1953.

[46] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. PMLR, 6861–6871.

[47] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based poi embedding for location-based recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 15–24.

[48] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning Fair Representations. In *Proceedings of the 30th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 28)*, Sanjoy Dasgupta and David McAllester (Eds.). PMLR, Atlanta, Georgia, USA, 325–333.

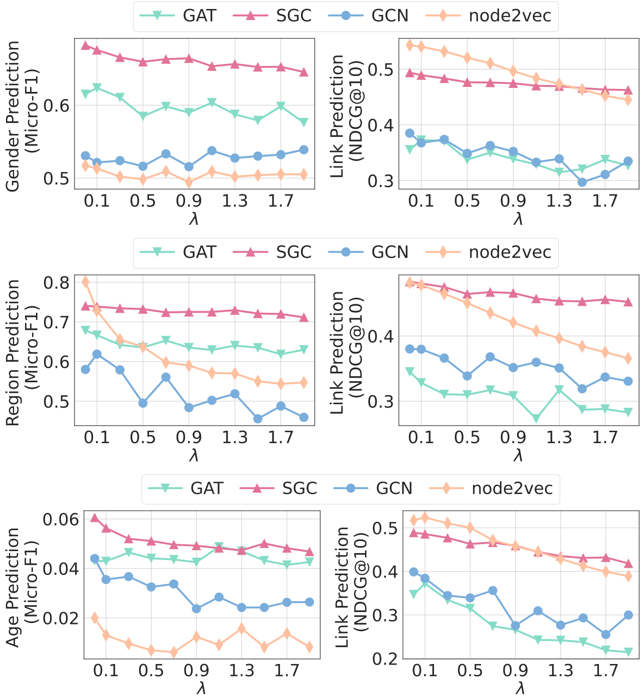


Figure 6: Unbiasedness and utility trade-off using different regularization weights on UGE-C (x-axis). The left columns shows unbiasedness (attribute prediction), and the right columns shows utility (link prediction).

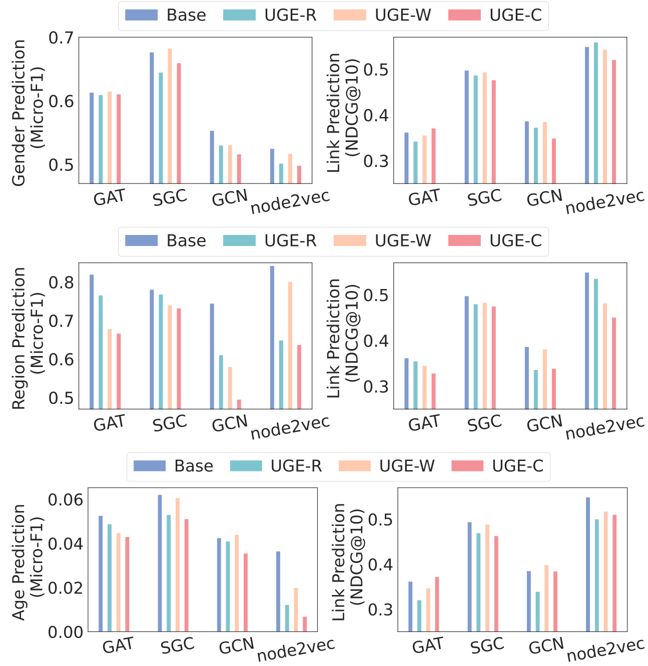


Figure 7: Comparison among our proposed models on different embedding models. The left columns shows the unbiasedness (attribute prediction) and the right columns shows the utility (link prediction).

A EXPERIMENTAL SETTINGS

Here we introduce more details about the experiment setup and model configurations for reproducibility.

For GCN-type models (GCN, GAT, SGC), we use two convolutional layers with dimension $d_1 = 64$ and $d_2 = 16$. For node2vec, we set walk length to 1 which turns a general skip-gram loss to objective of the link prediction task. All the deep learning models are trained via Adam optimizer with step size 0.01 for 800 epochs, and we use a normalized weight decay 0.0005 to prevent overfitting. Our proposed UGE methods and the baseline CFC require a regularization weight to balance the task-specific objective and the debiasing effect. For CFC, we report the result with the regularization weight chosen from the set $\{1.0, 5.0, 10.0, 15.0, 25.0, 35.0, 45.0, 55.0, 65.0\}$, which finally is $\lambda = 55.0$. For UGE, we test $\{0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9\}$, and report the performance when $\lambda = 0.5$. The regularization term in Eq (14) is summed over all node pairs and can be costly to calculate. But empirically, M group pairs sampled uniformly in each round of model update, where M is around 10% of the number of node groups, can already yield promising results. For evaluating the unbiasedness of the node embeddings, we use implementations from scikit-learn [34] for classifier training and evaluating Micro-F1.

B RESULTS

In Appendix B.1, we include additional experiment results to report the trade-off between unbiasedness and utility on the complete set of embedding models on Pokec-z. In Appendix B.2, we show a complete comparison among our proposed instances of unbiased graph embedding UGE-W, UGE-R and UGE-C. In Appendix B.3, we investigate the influence of the regularization weight on the complete set of embedding models.

B.1 Additional Analysis on Undebiasedness

Table 3 summarizes the debiasing and utility performance of the proposed method and baselines when using four graph neural networks on Pokec-z. Each line of attribute prediction result is followed by the corresponding performance on link prediction. Generally, UGE-W achieves the best link prediction performance and UGE-R has better debiasing effect. Combining UGE-W with UGE-R produces UGE-C with better trade-off.

B.2 Ablation Study

Figure 7 presents the performance of three proposed model (UGE-W, UGE-R and UGE-C) applied to four graph neural networks (GAT, SGC, GCN and node2vec). We can clearly observe that in most cases UGE-R has better debiasing effect compared with UGE-W, while UGE-W can better maintain the utility for downstream link prediction task. UGE-C as the combination of them indeed makes the best of the both designs.

B.3 Unbiasedness-Utility Tradeoff in UGE

In addition to Section 6.4 where we only showed the effect of regularization weight on Pokec-z with GAT as the embedding model,

Table 3: The prediction performance of node embeddings learned on Pokec-z using four graph neural networks as embedding models. In each row, we use bold to mark the best debiasedness on attribute prediction or utility on link prediction.

Dataset	Embedding Model	Prediction Target	No Debiasing	Debiasing Method					Random
				Fairwalk	CFC	UGE-W	UGE-R	UGE-C	
Pokec-z	GAT	Gender (Micro-F1)	0.6232	0.6135	0.5840	0.6150	0.6094	**0.5747**	0.4921
		Link (NDCG@10)	0.3618	0.3280	0.2757	**0.3554**	0.3422	0.3376	0.0570
		Region (Micro-F1)	0.8197	0.8080	0.7217	0.6784	0.7660	**0.6356**	0.4966
		Link (NDCG@10)	0.3618	0.3287	0.2757	0.3451	**0.3547**	0.3098	0.0570
		Age (Micro-F1)	0.0526	0.0522	0.0498	0.0431	0.0545	**0.0429**	0.0007
		Link (NDCG@10)	0.3618	0.3122	0.2757	0.3471	0.3205	**0.3718**	0.0570
	SGC	Gender (Micro-F1)	0.6766	0.6631	**0.6520**	0.6822	0.6531	0.6596	0.4921
		Link (NDCG@10)	0.4975	0.4461	0.4011	**0.4938**	0.4850	0.4765	0.0570
		Region (Micro-F1)	0.7806	0.7820	**0.7150**	0.7402	0.7680	0.7323	0.4966
		Link (NDCG@10)	0.4975	0.4460	0.4011	**0.4832**	0.4799	0.4644	0.0570
		Age (Micro-F1)	0.0621	0.0662	0.0654	0.0606	0.0529	**0.0510**	0.0007
		Link (NDCG@10)	0.4975	0.4461	0.4011	**0.4889**	0.4694	0.4630	0.0570
	GCN	Gender (Micro-F1)	0.5532	0.5589	0.5493	0.5306	0.5301	**0.5162**	0.4921
		Link (NDCG@10)	0.3865	0.2807	0.3836	**0.3851**	0.3727	0.3488	0.0570
		Region (Micro-F1)	0.7445	0.7616	0.7693	0.5800	0.6105	**0.4951**	0.4966
		Link (NDCG@10)	0.3865	0.2807	**0.3836**	0.3801	0.3360	0.3386	0.0570
		Age (Micro-F1)	0.0425	0.0416	0.0391	0.0439	0.0409	**0.0324**	0.0007
		Link (NDCG@10)	0.3865	0.2807	0.3836	**0.3987**	0.3550	0.3391	0.0570
	node2vec	Gender (Micro-F1)	0.5248	0.5347	0.5137	0.5171	**0.4949**	0.4982	0.4921
		Link (NDCG@10)	0.5491	0.5120	**0.5496**	0.5430	0.5463	0.5206	0.0570
		Region (Micro-F1)	0.8423	0.8462	0.8423	0.8012	0.6490	**0.6372**	0.4966
		Link (NDCG@10)	0.5491	0.5120	**0.5496**	0.4816	0.5354	0.4506	0.0570
		Age (Micro-F1)	0.0365	0.0404	0.0365	0.0200	0.0122	**0.0068**	0.0007
		Link (NDCG@10)	0.5491	0.5120	**0.5496**	0.5173	0.5439	0.5002	0.0570

we now include a complete analysis on unbiasedness and utility trade-off across embedding models in Figure 6. It clearly shows a trade-off: as the weight increases, we obtain a stronger debiasing effect with a cost of the utility on link prediction.