# Towards Fair Representation Learning in Knowledge Graph with Stable Adversarial Debiasing

Yihe Wang
*Department of Computer Science*
*University of North Carolina at Charlotte*
*ywang145@uncc.edu*

Mohammad Mahdi Khalili
*Yahoo! Research, USA*
*Ohio State University*
*mahdi.khalili@yahooinc.com*

Xiang Zhang
*Department of Computer Science*
*University of North Carolina at Charlotte*
*xiang.zhang@uncc.edu*

2022 IEEE International Conference on Data Mining Workshops (ICDMW) | 979-8-3503-4609-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICDMW58026.2022.00119

***Abstract*—With graph-structured tremendous information, Knowledge Graphs (KG) aroused increasing interest in academic research and industrial applications. Recent studies have shown demographic bias, in terms of sensitive attributes (e.g., gender and race), exist in the learned representations of KG entities. Such bias negatively affects specific populations, especially minorities and underrepresented groups, and exacerbates machine learning-based human inequality. Adversarial learning is regarded as an effective way to alleviate bias in the representation learning model by simultaneously training a task-specific predictor and a sensitive attribute-specific discriminator. However, due to the unique challenge caused by topological structure and the comprehensive relationship between knowledge entities, adversarial learning-based debiasing is rarely studied in representation learning in knowledge graphs. In this paper, we propose a framework to learn unbiased representations for nodes and edges in knowledge graph mining. Specifically, we integrate a simple-but-effective normalization technique with Graph Neural Networks (GNNs) to constrain the weights updating process. Moreover, as a work-in-progress paper, we also find that the introduced weights normalization technique can mitigate the pitfalls of instability in adversarial debasing towards fair-and-stable machine learning. We evaluate the proposed framework on a benchmarking graph with multiple edge types and node types. The experimental results show that our model achieves comparable or better gender fairness over three competitive baselines on Equality of Odds. Importantly, our superiority in the fair model does not scarify the performance in the knowledge graph task (i.e., multi-class edge classification).**

## 1. Introduction

Representation learning on Knowledge Graph (KG), composed of entities(nodes) and relations(edges), has drew tremendous attention [1], [2], [3] and has been applied to a wide range of applications, such as information extraction [4], recommending systems [5], and semantic parsing [6]. With the diverse usage of KG in many areas, fairness in data mining in KG has become a considerable issue. A biased model may hurt specific groups, especially the underrepresented subpopulations (in terms of gender, age, race, ethnicity, etc.) [3]. The machine learning model represents training data, while some data is biased due to historical issues. Besides, the training algorithm and training process can also amplify such bias already present in the data [7]. Many researchers on algorithmic fairness try to define and remove the bias in existing machine learning applications. Adversarial learning is a widely used in-processing method for mitigating bias in models [8], which is called *adversarial debiasing*. The adversarial debiasing framework usually consisted of a predictor for task-based prediction and a discriminator to recognize the sensitive attributes. For example, we want to predict the opportunity of university admission (i.e., target variable) given a student's application materials (i.e., input variable). In this process, we want to prevent the opportunity of admission affected by demographic information such as gender (i.e., *sensitive attributes*). The predictor network tries to predict the target variable given the input variable. In contrast, the discriminator tries to predict sensitive attributes given the underlying representation in the predictor. The goal of adversarial debiasing is to maximize the predictor's ability to predict the target of interest while minimizing the discriminator's ability to predict sensitive attributes. Ideally, the predictor is completely fair when the discriminator cannot predict sensitive attributes. However, the practical training of adversarial learning is challenging due to non-convergence and instability [9]. In other words, it is hard to train a model to always converge into an equilibrium that both have a good performance on target predicting while keeping a good fairness metric.

Adversarial learning, including adversarial debiasing, is inherently unstable since two competing networks are training simultaneously. To solve this problem, some practical techniques have been proposed to solve this instability of adversarial debiasing. It's found skew data distribution of sensitive attributes largely exacerbated the training instability [10]. For example, a machine learning model trained on a dataset with 80% males and 20% females will probably be unfair in terms of gender. [10] preprocessed the training data by manually choosing a subset dataset with a balanced distribution of the sensitive attributes. The results are much more stable compared to no preprocessing training, however, with a great sacrifice to performance. [11] forced the training

2375-9259/22/$31.00 ©2022 IEEE
DOI 10.1109/ICDMW58026.2022.00119

process to converge into a stable line by decreasing the learning rate of the predictor with a step counter. However, this method is very sensitive to parameter tuning, and it is hard to guarantee that the model converges into a point with both good performance and fairness metrics among different runnings.

To address the unfairness and instability in knowledge graph representation learning, we propose a framework with an adversarial debiasing module while regularizing the weights in the training process by spectral normalization. On the one hand, integrating with an adversarial debiasing module (i.e., predictor) can effectively reduce the sensitive information leakage in representation learning. On the other hand, adversarial training suffers from instability and non-converge issue due to its intrinsic characteristics (e.g., gradient exploding and gradient vanishing) [12]. We apply spectral-based weights regularization on the weights in graph message passing to achieve the trade-off between equality and stability. We evaluate the proposed framework on a simple yet popular knowledge graph in a link prediction task and assess the performance, equality, along with stability. Compared to three competitive baselines, our framework achieves superior model fairness with comparable (if not better) performance and more stable results.

In summary, we list the key contributions in this work as follow:

- To the best of our knowledge, this work is in the first batch of studies addressing the adversarial debiasing instability in knowledge graph data mining.
- We introduce a simple-but-effective weight normalization to smooth the training of graph representation learning, achieving the win-win of model fairness and training stability.
- We evaluate the proposed framework in a link prediction task over a benchmarking knowledge graph Compared with three baselines, our framework obtained better fairness (Equality of Odds) and higher stability without scarifying task performance.

The remaining of this paper is organized as follows. Section 2 describes related works on fairness, adversarial learning, and stability. Section 3 introduces the preliminary of knowledge graph and machine learning fairness, and we define our problem formulation in this section. In section 4, we describe the structure of our new method for fair and stable adversarial debiasing in the knowledge graph. We put the experiment setup and results in section 5. We discuss our limitations in section 6. Finally, section 7 presents the conclusion.

## 2. Related Work

### 2.1. Fairness in Machine Learning

Many definitions have been proposed to measure how fair a model is. The topic of machine learning fairness is rapidly developing which leads to a wide range of metrics. A model satisfies Demographic Parity if the model is independent of sensitive attributes [13]. A model satisfied Equality of Odds if the model is conditional independent of sensitive attributes given the ground truth label [14]. Equality of Opportunity is similar to Equality of Odds. A model is conditional independent of sensitive attributes given target equals to some specific value [15]. However, most of these metrics are considered in binary classification. [16] presented an approximate unfairness measurement that extends the demographic parity definition to multi-class classification. Counterfactual fairness is fairness definition based on causal inference to make fair decision towards individual [17]. [18] proved that removing the sensitive attributes in training data will not generate a fair model since some other attributes may correlate with the sensitive attributes. Methods of calibrating biased models are classified into three types: pre-processing [19] [20], in-processing [21] [22], and post-processing [23] [24].

### 2.2. Adversarial Learning

Adversarial learning is well known for its wide use in computer vision for generating realistic images [12] [25] [26] [27]. It is also a very effective and popular in-processing method to increase the fairness of a machine-learning model [28]. [10] applied adversarial learning to fair classification to achieve equality of opportunity. [11] is the first work to prove the correctness of achieving demographic parity, equality of odds, and equality of opportunity in adversarial debiasing. They also imported a novel idea that using a projection term in training. [29] advocated adversarial learning for fair and transferable representations. [30] designed a method that makes minibatch samples as diverse as possible to boost the effectiveness of adversarial debiasing. [31] applied adversarial debiasing on counterfactual fairness. [32] [2] [33] used adversarial debiasing to eliminate the bias in GNNs. [3] analyzed potential bias in knowledge graph and applied adversarial debiasing to mitigate bias in knowledge graph embeddings. The difference between [3] and our work is that they assume model is perfectly fair when discriminator reach random prediction on protected attributes. This assumption is weak in many cases. In this work, we employ equality of odds as fairness metrics to measure the exact fairness level of our model.

### 2.3. Stability

The definition of stability is different in literature. For example, stability definition in [34] [35] refers to model robustness with respect to variations in the training dataset; stability in [36] refers to robust to structural perturbations of a graph; [37] [38] focuses on the stability of hyperparameter changes in GANs. [9] [2] [11] defines the stability as the stable converge procedure and small fluctuation after converge in adversarial debiasing. [2] tries to solve the non-convergence issue with a covariance constraint, and [11] forces the model to converge with decreasing learning rate. We refer the stable convergence to *intra-running stability*. In contrast, we consider *inter-running stability* by the stable

optimal across different runnings (in identical experimental setups). In our paper, we focus on the inter-running stability, the variance among different experiments.

# 3. Preliminary and Problem Formulation

## 3.1. Knowledge Graph

Knowledge Graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is a graph contains a set of nodes $v \in \mathcal{V}$ and edges $e \in \mathcal{E}$ of various types. We use $e_{uv}$ to denote an edge $e = r(u, v)$ which connects node u and node v. The nodes $u, v$ have node types $\beta_1, \beta_2$, respectively. The $\mathcal{B}$ denotes the set of node types: $\beta_1, \beta_2 \in \mathcal{B}$. We set $|\mathcal{B}| = 2$ for simplification while acknowledge that the set $\mathcal{B}$ can be easily expanded to multiple node types. We denote the set of edge relations/types as $\mathcal{R}$ where the specific edge type $r \in \mathcal{R} : \beta_1 \times \beta_2 \rightarrow \{true, false\}$. We denote the feature of node $v$ as $\boldsymbol{x}_v \in \mathcal{X}$ while the label of edge $e$'s as $y \in \mathcal{Y}$. We define the sensitive attributes $z_{uv} \in \mathcal{Z}$ of an edge equals to the sensitive attributes from its end nodes $u$ or $v$ (stored in node features $\boldsymbol{x}_u$ or $\boldsymbol{x}_v$). In this work, we consider to protect the sensitive attributes in edge prediction task in the context of knowledge graph representation learning.

## 3.2. Fairness

Suppose we have a classifier $f : \boldsymbol{x} \rightarrow y$ and a dataset in format $(\boldsymbol{x}, y, z)$. In which, $\boldsymbol{x}$ denotes the input variable, $z \in \{0, 1\}$ denotes the binary sensitive attribute, $y \in \{1, ..., k\}$ denotes the sample label. Generally, the $z$ is an demographic attribute in $\boldsymbol{x}$, such as gender, race, or zip code. We use equality of odds [15] as metrics for fairness evaluation. Here are the definitions.

**Definition 1.** ***(Equality of Odds)****. A classifier $f$ satisfies equality of odds if the predictions $\hat{y}$ and sensitive attributes $z$ are independent given label $y$. For the binary sensitive attribute $z$, the definition can be written as:*

$$P(\hat{y}|z = 0, y) = P(\hat{y}|z = 1, y) \tag{1}$$

As an intellectual innovation, we define the fairness measurement of equality of odds for multi-class classification that expanding the idea of [16] on demographic parity [39].

**Definition 2.** ***(Multi-class Fairness Measurement)****. We measure the approximate fairness of equality of odds. For binary sensitive attribute $z$ and multiple classes of $y$, the definition can be written as:*

$$\Delta_{eo} = \max_{i \in \{1,...,k\}} \left\{ \left( \left| P(\hat{y} = i|y = i, z = 0) - P(\hat{y} = i|y = i, z = 1) \right| \right) + \left( \left| P(\hat{y} \neq i|y \neq i, z = 0) - P(\hat{y} \neq i|y \neq i, z = 1) \right| \right) \right\} \tag{2}$$

where classifier $f$ is perfectly fair on binary sensitive attribute $z$ if $\Delta_{eo} = 0$.

## 3.3. Stability

In this work, we consider inter-running stability which is measured by the standard deviation. Particularly, we run the experiment multiple times with identical setups and calculate the standard deviation for all evaluation metrics such as accuracy and equality of odds. The instability of a machine learning model is mainly caused by three aspects: 1) different dataset split; 2) diverse parameter initialization; 3) varying converge points in the gradient descending (i.e., model optimization) in different runnings. The third factor is the key and unique challenge brought by adversarial debiasing. To evaluate the ability of our model in learning stable representations, we control variables in multiple experimental runnings. By *identical setups*, we mean that we fix the dataset split (i.e., exactly the same set of training/testing samples are used in different runnings) and random initialization states (i.e., the initialization of models parameters are identical). In such case, the inter-running variations are solely caused by the fundamental property of adversarial model.

## 3.4. Problem Formulation

Suppose we have a knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. All the notations are defined in section 3.1. We aim to learn a fair and stable function $f(\mathcal{G}, \mathcal{X}, \mathcal{Z}) \rightarrow \hat{\mathcal{Y}}$ for multi-class edge classification task, where $\hat{\mathcal{Y}}$ is a set of predicted labels for unlabeled edges. Our goal is to maximize model fairness (evaluated equality of odds $\Delta_{eo}$) and inter-running stability without scarifying model performance of $f$. **The smaller $\Delta_{eo}$ denotes the better fairness**. Thus, we minimize $\Delta_{eo}$ while maxing accuracy and AUROC.

# 4. Method

## 4.1. Overview

Figure 1 shows the framework of our method. We present the details of our method in this section. Our model is composed by three main components: a graph encoder $f_e$, a task predictor $f_t$, and fairness discriminator $f_d$. The graph encoder $f_e$ is a graph neural network that takes $\mathcal{G}$ as input and learns node embeddings through message passing, aggregating, and updating. Then for each edge, we concatenate the node embeddings of its head and tail nodes into an edge embedding. This edge embedding is used as input of task predictor $f_t$ to estimate edge label $y_{uv}$. To force $f_e$ and $f_t$ learning fair edge representation, we add a fairness discriminator $f_d$ to regularize the training of $f_e$ and $f_t$. Specially, $f_d$ receives $\boldsymbol{h}_u$ and predicts sensitive attributes $\hat{z}_{uv}$, where $\boldsymbol{h}_u$ is a node embedding that contains latent sensitive information. Loss of $f_d$ will be used to update the parameters in $f_e$ and $f_t$. The $f_d$ paves $f_e$ and $f_t$'s parameters to move in a direction that will *fool* $f_d$ when doing optimization with gradient descent. Furthermore, we apply spectral normalization on weights of $f_e$ to improve the

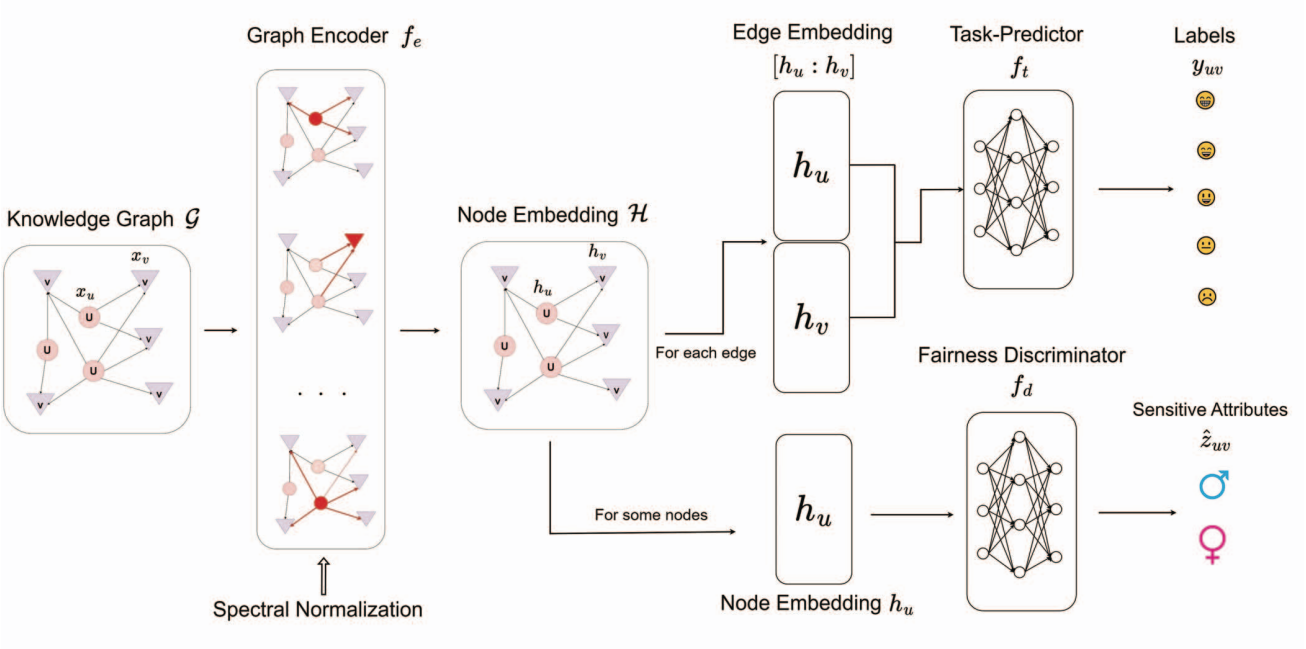


Figure 1: Framework of the proposed model. Graph encoder $f_e$ receives a knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ and generates the node embeddings $\boldsymbol{h}_u, \boldsymbol{h}_v \in \mathcal{H}$ for each node. Spectral normalization is applied on $f_e$'s weight. Every node $u, v \in \mathcal{V}$ has a corresponding node attributes $\boldsymbol{x}_u, \boldsymbol{x}_v$, respectively, where $u, v$ stand for different types of node. For each edge $e_{uv} \in \mathcal{E}$, we concatenate the node embeddings $\boldsymbol{h}_u, \boldsymbol{h}_v$ into an edge embedding $[\boldsymbol{h}_u : \boldsymbol{h}_v]$. Task predictor $f_t$ takes the edge embedding $[\boldsymbol{h}_u : \boldsymbol{h}_v]$ as input to predict edge labels $\hat{y}_{uv}$. Meanwhile, for some nodes that contains latent sensitive information, their node embedding $\boldsymbol{h}_u$ is used as input for fairness discriminator $f_d$ to predict sensitive attributes $\hat{z}_{uv}$, such as gender. Both task predictor and fairness discriminator are fully connected networks but can be straightforwardly switched to other neural architectures upon necessity.

fairness and stability. Spectral normalization is a method that was initially proposed to stabilize the training of discriminato in Generative Adversarial Networks [38]. Finally, $f_e$ and $f_t$ are the components we will keep for fair and stable knowledge graph edge label predicting. Notice that we use the lowercase letter to represent a single sample input, such as a node, an edge, or an embedding; we use the uppercase letter to denote a set of nodes, edges, or embeddings.

### 4.2. Graph Encoder

The graph encoder $f_e$ takes $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ as input and generates node embedding for each node considering topological context in knowledge graph. In this work, we adopt the widely-used SAGE [40] as backbone to construct graph encoder. Please note our framework can be straightforwardly expend to any graph neural network structures when building $f_e$. For each node $v \in \mathcal{V}$, there is a corresponding node features $\boldsymbol{x}_v$. Let $\boldsymbol{h}_v^l \in \mathcal{H}^l$ denotes the node embedding of node $v$ at the $l$-th layer of SAGE, where $\boldsymbol{h}_v^0 = \boldsymbol{x}_v$. Let $\mathcal{N}_v$ denotes the neighborhood of $v$; $W_v^l$ and $W_u^l$ denote weights while $\sigma^l$ denotes activation function of layer $l$. We do not apply activation function for the output layer. The updating of node embedding at layer $l$ (expect output layer) in SAGE can be written as:

$$\boldsymbol{h}_v^{l+1} = \sigma^l(W_v^l \cdot \boldsymbol{h}_v^l + W_u^l \cdot \frac{1}{|\mathcal{N}_v|} \sum_{u \in \mathcal{N}_v} \boldsymbol{h}_u^l) \tag{3}$$

Since the input is a knowledge graph (heterogeneous graph), we take the existing SAGE model and duplicates the message functions to work on each node and edge types individually [41]. In other words, we use different $W_v$ and $W_u$ for each node type.

In short, we denote the operation in graph encoder, i.e., the learning process of a node $v$, as:

$$\boldsymbol{h}_v = f_e(v, \mathcal{N}_v) \tag{4}$$

The set of node embedding $\mathcal{H}$ for all nodes in $\mathcal{G}$ can be written as:

$$\mathcal{H} = f_e(\mathcal{G}) \tag{5}$$

### 4.3. Task Predictor

The task predictor $f_t$ receives the edge embedding as input and predicts the edge label. For each edge $e_{uv} \in \mathcal{E}$, we build edge embedding $[\boldsymbol{h}_u : \boldsymbol{h}_v]$ for edge $e_{uv}$ by concatenating the corresponding source and target node embeddings.

We feed the edge embedding $[\boldsymbol{h}_u : \boldsymbol{h}_v]$ to $f_t$ and predict edge label $\hat{y}_{uv} \in \hat{\mathcal{Y}}$ through:

$$\hat{y}_{uv} = f_t([\boldsymbol{h}_u : \boldsymbol{h}_v]) \tag{6}$$

In this work, we use two layers of fully-connected layer as $f_t$ for simplify. The users are feel to modify it to suit their datasets and scenarios as needed.

At the dataset level, we calculate the set of predicted labels $\hat{\mathcal{Y}}$ for all edges in $\mathcal{G}$ as:

$$\hat{\mathcal{Y}} = f_t(\text{concat}(\mathcal{H})) \tag{7}$$

where *concat* means concatenating node embeddings into edge embeddings.

### 4.4. Fairness Discriminator

For each edge $e_{uv} \in \mathcal{E}$, we assume there exists latent sensitive information in at least one of its end node (take $u$ as an example in the following description). The fairness discriminator $f_d$ uses $\boldsymbol{h}_u$, the node embedding that contains latent sensitive information, as input to predict sensitive attributes. Usually, $f_d$ can predict sensitive attributes because $f_e$ and $f_t$ learn biased representation from knowledge graph. Ideally, $f_d$ would have very poor ability in sensitivity prediction if $f_e$ and $f_t$ are completely independent of sensitive attributes. In the adversarial training process, $f_e$ and $f_t$ try to learn a representation contains no information about sensitive attributes, while $f_d$ try to predict the sensitive attributes from the node embedding $\boldsymbol{h}_u$.

In order to measure the equality of odds in downstream procedures, we need to feed the edge label into discriminator. The predicted sensitive attributes $\hat{z}_{uv} \in \hat{\mathcal{Z}}$ for edge $e_{u,v}$ can be measured by:

$$\hat{z}_{uv} = f_d([\boldsymbol{h}_u : y_{uv}]) \tag{8}$$

where $\boldsymbol{h}_u$ and $y_{uv}$ are concatenated into one feature vector.

The set of predicted sensitive attributes $\hat{\mathcal{Z}}$ for all edges in $\mathcal{G}$ can be written as:

$$\hat{\mathcal{Z}} = f_d(\text{concat}(\mathcal{H}_u, \mathcal{Y})) \tag{9}$$

where *concat* means concatenating each $\boldsymbol{h}_u \in \mathcal{H}_u$ with its corresponding edge label $y_{uv} \in \mathcal{Y}$.

### 4.5. Spectral Normalization

Spectral normalization is a widely used technique applied on GANs to stabilize the training process and improve the image quality [38]. [37] proved it controls two important failure modes of GANs training: exploding and vanishing gradients.

Suppose there is a linear layer $g(\boldsymbol{h}) = W\boldsymbol{h}$ in a fully connected network. $W$ is the weight matrix. The spectral norm $\tau(W)$ of the matrix $W$ can be written as:

$$\tau(W) = \max_{\boldsymbol{h} \neq 0} \frac{||W\boldsymbol{h}||_2}{||\boldsymbol{h}||_2} \tag{10}$$

which is equivalent to the largest singular value of $W$. The $||\cdot||_2$ denotes L2 norm.

By definition, Lipschitz constant $||g||_{\text{Lip}}$ is given by [38]:

$$\begin{aligned} ||g||_{\text{Lip}} &= \sup_{\boldsymbol{h}} \tau(\nabla g(\boldsymbol{h})) = \sup_{\boldsymbol{h}} \tau(\nabla(W\boldsymbol{h})) \\ &= \sup_{\boldsymbol{h}} \tau(W) = \tau(W) \end{aligned} \tag{11}$$

where sup denotes supremum, and $\nabla$ denotes gradient.

Spectral normalization normalizes the weight matrix $W$ of $g$ by its spectral norm $\tau(W)$:

$$\widehat{W} = W/\tau(W) \tag{12}$$

where $\widehat{W}$ is the weight matrix after normalization. Now $\tau(\widehat{W}) = 1$ and $||g||_{\text{Lip}} = 1$. In general case, spectral normalization bounds the Lipschitz constant of a linear function by 1.

We implemented spectral normalization on the weight of $f_e$ to explore how it affects the fairness and stability for our edge label prediction model. Specifically, We normalize $W_v^l$ and $W_u^l$ in equation 3 for each layer $l$ of $f_e$:

$$\widehat{W}_v^l = W_v^l/\tau(W_v^l) \tag{13}$$

$$\widehat{W}_u^l = W_u^l/\tau(W_u^l) \tag{14}$$

where $\widehat{W}_v^l$ and $\widehat{W}_u^l$ are the weight matrix of layer $l$ after normalization.

### 4.6. Optimization

According to the definition of fairness metrics, $f_e$ and $f_t$ achieve perfect equality of odds if they are independent of sensitive attributes $\mathcal{Z}$ when given ground truth label $\mathcal{Y}$. We follow the implementation of adversarial debiasing as demonstrated in [11]. Equality of odds can be achieved by giving $f_d$ the edge label $\mathcal{Y}$ and $\mathcal{H}_u$.

We use cross-entropy loss $L$ for the training of graph encoder $f_e$ and task predictor $f_t$. The loss function for edge label $\mathcal{Y}$ prediction is defined as $L(f_t(f_e(\mathcal{G})), \mathcal{Y})$. To avoid the training instability caused by unbalanced data distribution with respect to sensitive attributes $\mathcal{Z}$, we use weighted cross entropy loss $L_w$ for training fairness discriminator $f_d$. Weight assigns to a label class is inversely proportional to a class percentage of the total number of samples. The loss function for sensitive attributes $\mathcal{Z}$ prediction is defined as $L_w(f_d(\mathcal{H}_u, \mathcal{Y}), \mathcal{Z})$.

The goal of optimization is to maximize $f_e$ and $f_t$'s ability in predicting edge labels while minimizing the $f_d$'s ability in predicting sensitive attributes. For equality of odds, the training process is denoted as:

$$\begin{aligned} &\arg\min_{f_e, f_t} L(f_t(f_e(\mathcal{G})), \mathcal{Y}) - \alpha L_w(f_d(\mathcal{H}_u, \mathcal{Y}), \mathcal{Z}) \\ &\qquad\qquad \arg\min_{f_d} L_w(f_d(\mathcal{H}_u, \mathcal{Y}), \mathcal{Z}) \end{aligned} \tag{15}$$

The training is iteratively performed between $f_e$, $f_t$ and $f_d$. The coefficient $\alpha$ is used to control how much $f_d$ *hurts* $f_e$ and $f_t$ if sensitive attributes are predictable by $f_d$. In other words, it controls the trade-off between edge label predicting ability and fairness.

TABLE 1. The Comparisons with Baselines. For equality of odds difference, the smaller the better; for inter-running statbility, the smaller the better.

| Final Result (Test result of final epoch) | | | | |
|---|---|---|---|---|
| Methods | Accuracy (%) | AUROC (%) | Equality of Odds Difference $\Delta_{eo}(\%)$ | Inter-running Stability (%) |
| EP | $40.11 \pm 1.59$ | $56.33 \pm 2.23$ | $10.29 \pm 1.76$ | 1.76 |
| AD | $41.74 \pm 0.28$ | $57.74 \pm 2.99$ | $8.42 \pm 1.31$ | 1.31 |
| WN-AD | $40.93 \pm 0.85$ | $57.07 \pm 1.70$ | $7.74 \pm 2.45$ | 2.45 |
| SP-AD | **41.53 ± 0.16** | **58.67 ± 2.49** | **7.92 ± 0.73** | **0.73** |
| Best Result (Best fairness result when accuracy $> 40\%$) | | | | |
| Methods | Accuracy (%) | AUROC (%) | Equality of Odds Difference $\Delta_{eo}(\%)$ | Inter-running Stability (%) |
| EP | – | – | – | – |
| AD | $41.64 \pm 0.19$ | $57.88 \pm 3.39$ | $6.07 \pm 0.79$ | 0.79 |
| WN-AD | – | – | – | – |
| SP-AD | **41.50 ± 0.23** | **58.85 ± 2.32** | **5.32 ± 0.39** | **0.39** |

## 5. Experiments

In this section, we provide the details of dataset, experimental setup, baselines, and comparison results. We use Pytorch Geometric (PYG) [41] in model implementation.

### 5.1. Dataset

**5.1.1. Knowledge graph.** We run experiments on the MoiveLens 100K dataset [42] which is benchmark in recommendation system. Here we use it to build a simple but typical knowledge graph. In the knowledge graph, we have two types of nodes and five types of edges. Each edge connects a user node and a movie node, denoting the user's rating to a movie. The five types of edges correspond to five ratings (1-point to 5-point). In summary, our graph contains 2625 nodes (including 943 users and 1682 movies) and 100,000 edges (rating score 1, 2, 3, 4, and 5 for the five edge types, respectively. Each user has rated at least 20 movies. Besides, there is also some demographic information for the users, such as age, gender, occupation, and zipcode. The feature dimension for the user node is 815, and for the movie node is 403 after processing. We used the *HeteroData* class in the PYG package to store KG.

**5.1.2. Data Preprocessing.** The following are the details for dataset cleaning and preprocessing. We clearn movie attributes and user attributes before storing them as nodes in HeteroData class. For movie attributes, *SentenceTransformer* with model name "all-MiniLM-L6-v2" was used to convert the *title* attribute into tensors. We use categorical attributes, occupation, zip code, binary attributes, gender, and dropped age attributes. Then we store all movie and user attributes tensors into HeteroData to represent movie and user nodes information.

The relations between user and movie nodes are ratings from 1 to 5 points. We dropped the timestamp attributes and stored the relations as edges into HeteroData. One edge is in the form of two indexes (from user to movie) and a label(rating). The user and movie ID in original dataset starts from 1, while our index to store user and movie nodes starts from 0. We need to shift the user and movie ID by 1 when we store ratings as edges into HeteroData.

For easier message passing, we add reverse edges by using function *ToUndirected()* in *torch_geometric.transforms*. We deleted the label for these reverse edges since they are useless. *RandomLinkSplit()* in *torch_geometric.transforms* is used to split the data into training and test set in 8:2 ratio. We fix the random split seed by using *torch_geometric.seed_everything()* function and set the seed to 2. To measure the gender bias in our model, we need to further split the test set into male_test set and female_test set by gender attribute. Gender attribute is stored in the user nodes, and we need to look at the user node on each edge. For example, suppose we have an edge (27, 133) with label 3, which means $27_{th}$ user node rates 3 on $133_{th}$ movie node. We manually put this edge into the male test set if the user node gender is male, and vice versa. Finally, we have a knowledge graph with 943 user nodes with 815 feature dimensions and 1682 movie nodes with 403 feature dimensions. There are 80000, 14954, and 5046 edges in training, male test, and female test set.

### 5.2. Experimental setup

**5.2.1. Baselines.** We compared our method with three baselines: edge predictor (EP), adversarial debiasing (AD), Weighted Normalization on $f_e$ (WN-AD).

- **EP** : This is a knowledge graph edge predictor. It only has graph encoder $f_e$ and task predictor $f_t$. In other words, it is a normal data mining method over knowledge graph which is not equipped with debiasing strategy.
- **AD**: This is an adversarial debiasing architecture includes a predictor for prediction and a discriminator for bias removal. It contain graph encoder $f_e$, task predictor $f_t$ and fairness discriminator $f_d$.
- **WN-AD**: Besides all we have in AD, this method apply weighted normalization on $f_e$.

The difference between our model SP-AD and WN-AD is that we apply spectral normalization while WN-AD uses weighted normalization. We want to see the comparisons with other commonly used normalization methods. Note that the comparison with baseline also can serve as ablation study of our framework.

**5.2.2. Evaluation Metrics.** We focus on three aspects of the model: performance, fairness and stability.

- **Performance metrics**: We report accuracy and AUROC (Area Under ROC Curves) to evaluate the performance of edge label prediction. The higher accuracy and AUROC denote better performance.
- **Fairness metrics**: We measure the fairness through equility of odds as defined in section 3.2. The smaller $\Delta_{eo}$, the more fair model. The model is perfectly fair when $\Delta_{eo} = 0$.
- **Stability metrics**: We assess the inter-running stability of fairness by standard deviation across multiple independent runnings. In particular, we run the experiment ten times with identical experimental results (the data split, hyper-parameters, and parameter initialization are fixed across different runnings) and calculate the standard deviation for $\Delta_{eo}$.

**5.2.3. Implementation Details.** We use two layers of SAGE to construct the graph encoder $f_e$. Since we are training on a knowledge graph with two types of nodes, we use lazy initialization that sets the input tuple as (-1,-1) to deal with different dimensions of node attributes. Both task predictor $f_t$ and fairness discriminator $f_d$ are constructed by two layers of fully-connected neural layers. For $f_t$, the input dimension is two times of the dimension in $f_e$ because each edge embedding is concatenated by two node embeddings. The output of edge predictor is a 5-class label $\hat{y}_{uv} \in \{1, ..., 5\}$, so the output dimension is 5. For $f_d$, it needs to take the node embedding $\boldsymbol{h}_u$ and $y$ as input to calculate equality of odds. Since edge labels are encoded in one-hot format, the input dimensions should be the dimension of $\boldsymbol{h}_u$ plus five. We use ReLU as activation function in all hidden layers. We apply spectral normalization on all layer's weights in $f_e$. In preliminary experiments, we tried to apply spectral normalization to $f_t$ and $f_d$, but resulting to performance degradation.

In table 2, SG stands for SAGE layer, and FC stands for fully connected layer, EO stands for equality of odds, and SP stands for spectral normalization.

TABLE 2. Model Architecture

| Components | Architecture |
|---|---|
| $f_e$ | SP(SG((-1,-1), 32)), ReLU, SP(SG((-1,-1),32)) |
| $f_t$ | FC(2*32, 32), ReLU, FC(32,5), Softmax |
| $f_d$ | FC(32+5, 32), ReLU, FC(32,1), Sigmoid |

To fix the parameter initialization, we set the random seed to 0 by function *torch.manual_seed()*. We use Adam optimizer to minimize the loss for $f_e$, $f_t$, and $f_d$. The learning rates for $f_e$ and $f_t$ are set as 0.005 while the learning rate for $f_d$ is 0.001. We fix the hyper-parameter $\alpha$ to 0.5. We iteratively minimize the loss of $f_e$, $f_t$ and $f_d$. In particular, we fix $f_d$ while minimizing **one** epoch for $f_e$ and $f_t$, then fix $f_e$ and $f_t$ while minimizing **twenty** epochs for $f_d$. We assume the model is converged into equilibrium when test accuracy does increase. The number of required epochs to converge vary when different methods and fairness metrics are applied. In this case, we choose the number of epochs that guarantee converge of model (either our model or baselines). In other words, the stopping criterion is precisely defined for each model for fair comparison. Be aware that the number of epochs here refers to the epoch number that minimizes $f_e$ and $f_t$; the epoch number of $f_d$ should be $20\times$, as mentioned before. As a result, we set 2000, 2000, 5000, 4000 epochs for EP, AD, WN-AD and SP-AD. The epoch number is the only variant among the setups of different methods.

## 5.3. Results

Table 1 shows the results for overall comparisons. SP-AD is our proposed method: adversarial debiasing with spectral normalization. We report results from two aspects: the final result and the best fairness result. The final results mean the test results using the model saved at the *last training epoch*. For the best fairness results, the model is saved under early-stop criterion. In specific, we store the model parameters that achieve an accuracy larger than 40% while obtaining the best $\Delta_{eo}$ among all training epochs. We choose $40\%$ as the threshold because the best accuracy among all methods we achieved is around $42\%$. We think $2\%$ of accuracy sacrifice is an acceptable value for the trade-off between performance and fairness. Recall that we run the experiment 10 times with all setups the same except training epochs, and each running has a final result and the best fairness result. We calculate the standard deviation for all results to calculate the inter-running stability.

For the final result, our method achieves 0.73 inter-running stability, which is the best result compared with baselines (i.e., smaller than other baselines). Our method beats EP on all the metrics, including accuracy, AUROC, $\Delta_{eo}$, and inter-running stability. Our method's $\Delta_{eo}$ is better than AD on both value and stability while keeping almost the same accuracy and even better AUROC. The $\Delta_{eo}$ of WN-AD is slightly better ($0.18\%$) than our method, but it strongly sacrifices the accuracy and $\Delta_{eo}$ stability. Our method obtains $0.6\%$ higher accuracy and $1.72\%$ less instability than WN-AD.

In terms of best result, we only report for AD and SP-AD because the other two methods do not always have a result with accuracy $> 40\%$. Some running results only have accuracy around $38.5\%$ even when the model converges. We believe the reason is the local minimum issue. In other words, AD and SP-AD are better methods to achieve accuracy than EP and WN-AD. For comparison between AD and SP-AD, our method has better results on AUROC, $\Delta_{eo}$, and inter-running stability while keeping almost the same accuracy level.

# 6. Discussion

As a work-in-progress paper, our method has some limitations. First, we only test our method on Movielens 100K, which is a simple knowledge graph that only has two types of nodes and five types of edges. The unfairness

in this knowledge graph is not that huge, even with a pure classifier (no fairness calibration). It is hard to demonstrate the effectiveness of our method with smaller unfairness. Second, we only compare with one baseline related to weight normalization. It's a promising future work that comparing with more baselines with some GNN normalization methods (such as GraphNorm [43]). Third, we consider a single sensitive attribute (gender) in this work, while multiple biases could be involved simultaneously (e.g., gender and age and ethnicity) in the real world.

## 7. Conclusion

In this paper, we present a high-performance and stable method for unbiased knowledge graph representation learning. We use adversarial learning to remove sensitive information in an edge classification task. Our model consists of a node encoder, an edge decoder, and a fairness discriminator. We learn a representation jointly using node encoder and edge decoder while the fairness discriminator can hardly predict sensitive attributes based on learned representation. Aiming to alleviate the inter-running instability of fairness, we integrate spectral normalization with graph neural network to regularize the weights in messaging passing and integration. Comprising with several baselines, our experimental results show that our framework achieves good fairness and better stability while keeping comparable (if not better) edge classification performance. For future work, we will work on a model that also guarantees good intra-running stability on fairness, which basically means the converge curve of the equality of odds will be more smooth within the same running.

## References


[1] X. Yuan, C. Xu, P. Li, and Z. Chen, "Relational learning with hierarchical attention encoder and recoding validator for few-shot knowledge graph completion," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022, pp. 786–794.

[2] E. Dai and S. Wang, "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*, 2021, pp. 680–688.

[3] M. Arduini, L. Noci, F. Pirovano, C. Zhang, Y. R. Shrestha, and B. Paudel, "Adversarial learning for debiasing knowledge graph embeddings," *arXiv preprint arXiv:2006.16309*, 2020.

[4] L. Dietz, A. Kotov, and E. Meij, "Utilizing knowledge graphs for text-centric information retrieval," in *The 41st international ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 1387–1390.

[5] B. Shao, X. Li, and G. Bian, "A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph," *Expert Systems with Applications*, vol. 165, p. 113764, 2021.

[6] L. Heck and H. Huang, "Deep learning of knowledge graph embeddings for semantic parsing of twitter dialogs," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2014, pp. 597–601.

[7] Y. R. Shrestha and Y. Yang, "Fairness in algorithmic decision-making: Applications in multi-winner voting, machine learning, and recommender systems," *Algorithms*, vol. 12, no. 9, p. 199, 2019.

[8] M. Du, F. Yang, N. Zou, and X. Hu, "Fairness in deep learning: A computational perspective," *IEEE Intelligent Systems*, vol. 36, no. 4, pp. 25–34, 2020.

[9] D. Saxena and J. Cao, "Generative adversarial networks (gans) challenges, solutions, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–42, 2021.

[10] A. Beutel, J. Chen, T. Doshi, H. Qian, A. Woodruff, C. Luu, P. Kreitmann, J. Bischof, and E. H. Chi, "Putting fairness principles into practice: Challenges, metrics, and improvements," in *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 2019, pp. 453–459.

[11] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 335–340.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[13] T. Calders and S. Verwer, "Three naive bayes approaches for discrimination-free classification," *Data mining and knowledge discovery*, vol. 21, no. 2, pp. 277–292, 2010.

[14] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[15] C. S. Crowson, E. J. Atkinson, and T. M. Therneau, "Assessing calibration of prognostic risk scores," *Statistical methods in medical research*, vol. 25, no. 4, pp. 1692–1706, 2016.

[16] C. Denis, R. Elie, M. Hebiri, and F. Hu, "Fairness guarantee in multi-class classification," *arXiv preprint arXiv:2109.13642*, 2021.

[17] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," *Advances in neural information processing systems*, vol. 30, 2017.

[18] K. Lum and J. Johndrow, "A statistical framework for fair predictive algorithms," *arXiv preprint arXiv:1610.08077*, 2016.

[19] B. d'Alessandro, C. O'Neil, and T. LaGatta, "Conscientious classification: A data scientist's guide to discrimination-aware classification," *Big data*, vol. 5, no. 2, pp. 120–134, 2017.

[20] Y. Qiang, C. Li, M. Brocanelli, and D. Zhu, "Counterfactual interpolation augmentation (cia): A unified approach to enhance fairness and explainability of dnn," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, LD Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization*, vol. 7, 2022, pp. 732–739.

[21] M. Wan, D. Zha, N. Liu, and N. Zou, "In-processing modeling techniques for machine learning fairness: A survey," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2022.

[22] M. Wick, J.-B. Tristan *et al.*, "Unlocking fairness: a trade-off revisited," *Advances in neural information processing systems*, vol. 32, 2019.

[23] P. K. Lohia, K. N. Ramamurthy, M. Bhide, D. Saha, K. R. Varshney, and R. Puri, "Bias mitigation post-processing for individual and group fairness," in *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2019, pp. 2847–2851.

[24] P. Nandy, C. Diciccio, D. Venugopalan, H. Logan, K. Basu, and N. El Karoui, "Achieving fairness via post-processing in web-scale recommender systems," in *2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022, pp. 715–725.

[25] Z. Li, P. Xia, R. Tao, H. Niu, and B. Li, "A new perspective on stabilizing gans training: Direct adversarial training," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2022.

[26] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017.

[27] A. Lattas, S. Moschoglou, S. Ploumpis, B. Gecer, A. Ghosh, and S. P. Zafeiriou, "Avatarme++: Facial shape and brdf inference with photorealistic rendering-aware gans," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 01, pp. 1–1, 2021.

[28] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[29] D. Madras, E. Creager, T. Pitassi, and R. Zemel, "Learning adversarially fair and transferable representations," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3384–3393.

[30] T. Adel, I. Valera, Z. Ghahramani, and A. Weller, "One-network adversarial fairness," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 2412–2420.

[31] V. Grari, S. Lamprier, and M. Detyniecki, "Adversarial learning for counterfactual fairness," *Machine Learning*, pp. 1–23, 2022.

[32] F. Masrour, T. Wilson, H. Yan, P.-N. Tan, and A. Esfahanian, "Bursting the filter bubble: Fairness-aware network link prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 841–848.

[33] H. Liu, N. Zhao, X. Zhang, H. Lin, L. Yang, B. Xu, Y. Lin, and W. Fan, "Dual constraints and adversarial learning for fair recommenders," *Knowledge-Based Systems*, vol. 239, p. 108058, 2022.

[34] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth, "A comparative study of fairness-enhancing interventions in machine learning," in *Proceedings of the conference on fairness, accountability, and transparency*, 2019, pp. 329–338.

[35] L. Huang and N. Vishnoi, "Stable and fair classification," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2879–2890.

[36] C. Agarwal, H. Lakkaraju, and M. Zitnik, "Towards a unified framework for fair and stable graph representation learning," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 2114–2124.

[37] Z. Lin, V. Sekar, and G. Fanti, "Why spectral normalization stabilizes gans: Analysis and improvements," *Advances in neural information processing systems*, vol. 34, pp. 9625–9638, 2021.

[38] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[39] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *Proceedings of the 3rd innovations in theoretical computer science conference*, 2012, pp. 214–226.

[40] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[41] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[42] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[43] T. Cai, S. Luo, K. Xu, D. He, T.-y. Liu, and L. Wang, "Graphnorm: A principled approach to accelerating graph neural network training," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1204–1215.