Debiasing graph neural networks for link prediction

Anonymous Authors[1]

Abstract

Graphs are a ubiquitous form of data that have been used for a variety of tasks. However, graph data can sometimes contain human biases, which makes their way into downstream systems. For example, a recommendation system for social networks may learn strong gender and ethnic homophily, making biased recommendation and influencing critical human decisions. In particular, identifying and removing biases are becoming notoriously challenging due to the increasing adoption of complex graph neural networks. Here, we propose a novel debiasing framework for graph neural networks. Many graph neural networks are trained on *contrastive learning* framework, in which a neural network is trained to discriminate between the given and randomly-generated networks. Bias is developed in graph neural networks because it is often a useful feature for the discrimination task. Our proposed approach introduces bias to the random networks so that it is not a useful feature for the discrimination task, thus preventing the bias from entering the neural networks. Our framework makes the bias model explicit, providing greater control and transparency. We evaluate our framework on link prediction and demonstrate that it substantially reduces bias while maintaining overall link prediction accuracy.

1. Introduction

The increasing reliance on algorithms in text and image processing has brought attention to human biases (Bender et al., 2021) and a critical need for fairness in the use of machine learning models (Bolukbasi et al., 2016; Bourli & Pitoura; Gen; Goo; Pages). Yet, another common and ubiquitous type of data—graphs—has received far less attention than they deserve. Graph data is not only the foundation of social and biological data (Newman) but ubiquitous across a huge range of computational problems (Jumper et al., 2021; Derrow-Pinion et al., 2021; Mirhoseini et al., 2021). For example, because social networks exhibit a strong homophily—the tendency for individuals to form connections with others who share similar characteristics such as gender, ethnicity, and culture (Yang, 2013)—AI systems trained on these social networks may make biased recommendations based on the homophily in high-stakes areas, such as professional networking that can influence hiring and promotions. For instance, it is well-known that the leadership roles in firms are dominated by white male population. *If* the recommendation engine of a professional networking service (e.g., LinkedIn) is optimized to predict the existence of actual professional acquaintance, the system is more likely to, from the homophily signal that it has learned from the existing data, recommend connections between white male professionals in the majority group, depriving opportunities from other populations and strengthening existing biases in the industry.

Let us demonstrate a potentially harmful consequence of network biases by using a small network of political books (Kunegis, 2013). This network consists of 105 books written around the time of 2004 US presidential election consisting of 441 edges (co-purchase relation) between books. Many AI systems hinge on graph embedding (Grover & Leskovec, 2016; Hamilton et al., 2017; Perozzi et al.; Veličković et al., 2017), where each node is represented as a point in space, with geometry reflecting the network structure. The network has a strong political discrimination, and so does the embedding generated by a standard graph embedding—LINE (Tang et al.) (Fig.1B). Recommendation systems based on the embedding may promote books with a specific ideology and reinforce echo chambers while sidelining other viewpoints, ultimately exacerbating political polarization.

A common debiasing approach is based on the manipulation of the input data and output embedding, e.g., balancing the training data (Khajehnejad et al.; Rahman et al.; Solaiman & Dennison) or removing embedding dimensions associated with bias (Bolukbasi et al., 2016; Ravfogel et al., 2020). However, these manipulation approaches may degrade embedding quality, fail to remove the bias, or even

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

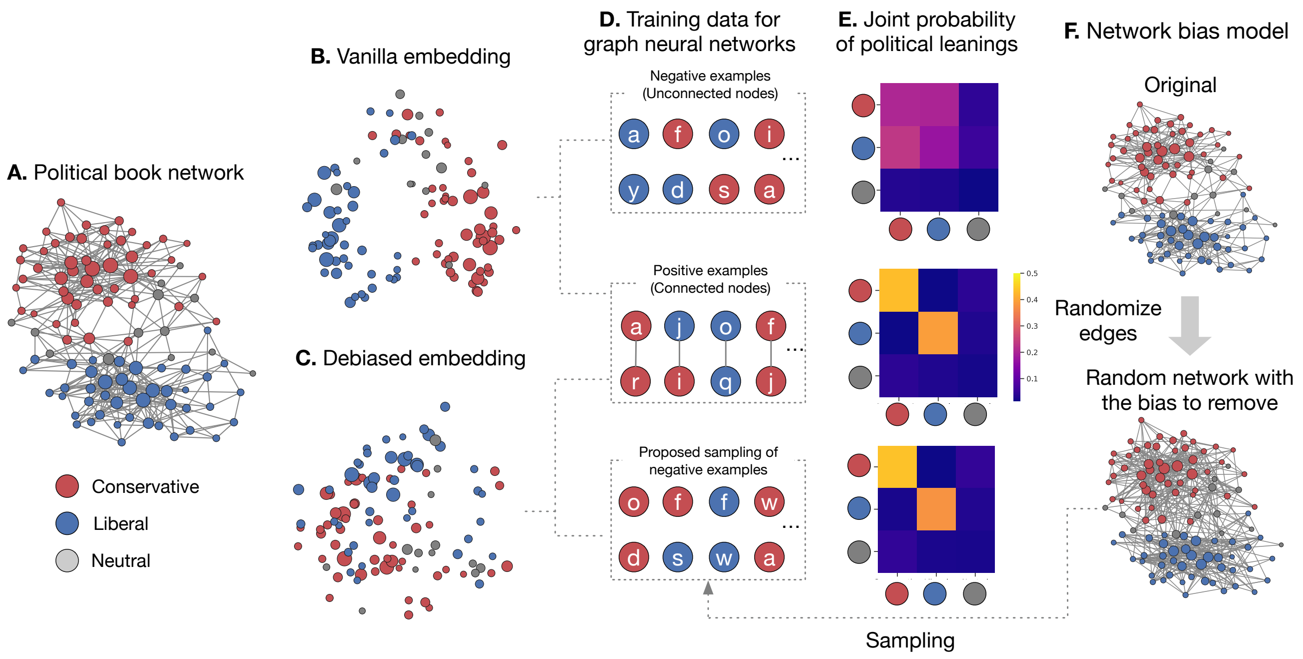

A. Political book network
B. Vanilla embedding
C. Debiased embedding
D. Training data for graph neural networks
Negative examples (Unconnected nodes)
Positive examples (Connected nodes)
Proposed sampling of negative examples
E. Joint probability of political leanings
F. Network bias model
Original
Randomize edges
Random network with the bias to remove
Sampling
Conservative
Liberal
Neutral

Figure 1. Proposed debiasing framework. **A.** The political book network consists of 105 nodes representing books, with colors representing the political leaning. **B.** LINE graph embedding trained with a set of pairs with/without edges. **C.** LINE graph embedding trained with the proposed debiasing framework. **E.** Joint probability distribution of political leaning. The negative examples for vanilla LINE have no political homophily, whereas the the positive edges have a strong homophily. LINE learns this misalignment and reflected it on the embedding. We prevent LINE from learning the bias by aligning the political homophily, which results in the embedding with less clear political discrimination. **F.** We sample the biased negative examples from a random network generated by a biased network model.

worse, introduce a new bias (Ravfogel et al., 2020; Bose & Hamilton; Edwards & Storkey, 2015). An alternative approach is adversarial learning, in which an adversarial model attempts to extract any biased features from the embedding, while the embedding is modified to be resistant to the adversary's attempts. However, the adversarial learning is often hindered by its high instability and sensitivity to hyperparameters (Xing et al., 2021).

We propose a simple yet effective framework for debiasing graph data that utilizes the inherent capacity of neural networks (Kojaku et al.). Our key idea is to use a *biased contrastive learning*, which trains a neural network to discriminate between actual data and random data. By introducing a specific bias into the random data, we ensure that sensitive attributes are not informative in the discrimination, *preventing biases from entering the neural network* (Fig. 1C). The proposed training framework can be applied to a wide range of models from simple graph embedding methods (e.g., DeepWalk and node2vec) to deep graph neural networks (e.g., GCN and GAT), offering a flexible and practical alternative to existing debiasing methods. We test our approach by using link prediction task, demonstrating that our proposed framework substantially reduces biases while maintaining high link prediction accuracy as compared to the prevailing debiasing methods.

2. Method

2.1. Networks

We assume that a network consists of N nodes and M edges. Each node i has a C-dimensional vector x_i representing the i's attributes. Each edge (i, j) between nodes i and j can be directed and have weight w_{ij}. We assume that the given network is weakly connected—every node is reachable from every other node when we ignore the direction of the edges.

2.2. Link prediction

Missing link prediction is a central task of graph representation learning and the basis of recommendation systems for social networks. We test graph neural networks with a standard benchmark of link prediction as follows. First, we randomly partition the set of edges in a given undirected network into a training set and a test set of almost an equal size while keeping the network constructed from the train edges being connected (Grover & Leskovec, 2016; Yoon et al., 2021). We ensure the connectedness of the test network by adding the edges in a minimum spanning tree to the train set (Grover & Leskovec, 2016; Yoon et al., 2021). Second, we generate negative edges—the pairs of unconnected nodes (i, j)—from the given network by sampling i and j uniformly at random. We generate the same number of negative examples as the number of test edges. Third, we train a graph embedding model using the network constructed with the train edge set. Fourth, for each test edge and negative edge, we calculate the likelihood of edge between two nodes by the dot similarity of the nodes' embedding vectors. We evaluate the likelihood of edge by using the area under curve of the receiver operating characteristics curve (AUC-ROC). We run the experiment five times with different random seeds.

2.3. Key idea

Graph neural network for link prediction is trained to differentiate the presence and absence of edges, a training framework called *contrastive learning* (Kojaku et al.; Dyer, 2014; Gutmann & Hyvärinen, 2010). Contrastive learning has been used not only for link prediction but also for a variety of tasks including natural language processing and image recognition. For example, in a face recognition task, a neural network learns a person's face x by contrasting it with a reference face x' of a randomly-sampled person. The neural network recognizes target face x by focusing on the *differences* to the reference face x'. Now, if we sample reference x' from the same ethnic group as target x, the neural network learns the differences within the group, rather than learning ethnic traits. This can prevent the neural network from developing ethnic bias.

In context of graph embedding, a positive example is an edge $x = (i, j)$ between nodes i and j. A negative example (or negative edge) is a pair of unconnected nodes $x' = (i', j')$ in a given network (Kojaku et al.). As in the case of face recognition task, a graph embedding model learns the *difference* between the positive edge x and negative edge x'. Our key idea is to generate the negative edges with the same biases so that the positive and negative edges are indistinguishable in terms of the biases, which prevents the graph embedding model from learning the biases.

Let us demonstrate our key idea by using the political book network. Consider the political orientations as a sensitive attribute, and we want remove them from the embedding. The political attribute correlates with the network structure, namely a book is likely to be connected with the one with the same political leaning, resulting in many positive edges formed by books with the same political leaning (Fig. 1E). Now, many graph neural networks including LINE are trained on negative edges that are sampled uniformly at random. Since the sampling of negative edges are independent of political leaning, the sampled negative edges do not have strong political homophily. This results in the misalignment of the positive and negative edges in terms of political homophily, which is then picked up by a graph embedding model as a useful feature for discrimination.

We prevent the model from using the sensitive attribute by generating negative examples with the same bias characteristics as the positive examples. To this end, we randomize the structure of a given network while preserving the joint probability of political leaning based on the maximum-entropy framework (Fig. 1F; see the Model of network bias section). This results in a random network that is indistinguishable from the given network in terms of the political assortativity (Fig. 1E), and the resulting graph embedding has no visible political separation accordingly (Fig. 1C).

2.4. Contrastive learning for debiasing

We focus on graph neural networks that generate a node embedding based on the network structure and node features. We express a graph neural network as a function ϕ parameterized by θ that produces an embedding of node i, i.e., $\phi(i;\theta) \in \mathbb{R}^{K\times 1}$. In link prediction, a graph neural network learns the probability $P(i, j)$ that an edge appears between two nodes (i, j), a common form being the softmax function, i.e.,

$$P(i,j) = \frac{1}{Z}\exp(\phi(i)^\top \phi(j)). \tag{1}$$

Variable Z is a normalization constant. Fitting the softmax function is notoriously difficult since Z is computationally demanding as it extends over all node pairs. Thus, several efficient estimation methods have been developed.

Noise contrastive learning (NCE) is an efficient estimation method for Eq. (1) (Gutmann & Hyvärinen, 2010). The key idea underlying NCE is to train a different computationally-cheaper model whose best parameter θ^* in terms of the likelihood being the same as Eq. (1). Operationally, NCE trains a logistic regression model by classifying a set of node pairs (i, j) into *connected* node pairs and randomly-sampled node pairs, i.e.,

$$P\left(Y_{ij}=1\right) := \frac{1}{1+\exp(-f(i,j)+\ln P_0(i,j))}, \tag{2}$$

where $Y_{ij} = 1$ if nodes i and j are connected, $Y_{ij} = 0$ if they are not, and $f(i, j) = \phi(i)^\top\phi(j)$ is the node similarity. It is shown that NCE is asymptotically unbiased for an exponential probability model given by (Gutmann & Hyvärinen, 2010; Dyer, 2014):

$$P(i,j) = \frac{1}{Z}\exp(f(i,j)), \tag{3}$$

which corresponds to the edge probability learned by a graph neural network given inEq. (1).

A key feature of NCE is that it is unbiased estimator agnostic to the choice of $P_0(i, j)$ because the effect of $P_0(i, j)$ is offset by $\ln P_0(i, j)$ in Eq. (2). Removing the offset gives rise to a powerful capacity of debiasing, as is demonstratd by (Kojaku et al.). Let us consider NCE without the offset

$\ln P_0(i,j)$, i.e., discriminating the connected and unconnected node pairs by

$$P\left(Y_{ij}=1\right) := \frac{1}{1+\exp(-\phi(i)^\top \phi(j))} \tag{4}$$

Because we drop $\ln P_0(i,j)$, Eq. (4) gives a biased estimate. To see the effec of the bias, we derive a model for which Eq. 4 is unbiased for by rewriting Eq. (4) in form of NCE as

$$P\left(Y_{ij}=1\right) := \frac{1}{1+\exp\left(-f'(i,j)+\ln P_0(i,j)\right]} \tag{5}$$

where $f(i,j) := \phi(i)^\top \phi(j) + \ln P_0(i,j)$. Since Eq. (2) is unbiased estimator for Eq. (3), Eq. (5) is asymptomatically unbiased for

$$\begin{aligned} P(i,j) &= \frac{1}{Z'}\exp(f'(i,j)) \\ &= \frac{1}{Z'}P_0(i,j)\exp(\phi(i)^\top \phi(j)). \end{aligned} \tag{6}$$

Equation (6) decomposes the probability of edge $P(i,j)$ into $P_0(i,j)$ and node similarity $\phi(i)^\top \phi(j)$. From a different perspective, noise distribution $P_0(i,j)$ serves as a *null model* for networks, accounting for trivial relationships of nodes. The node similarity captures the *residuals* from $P_0(i,j)$, reflecting the non-trivial relationships not explained by the null model. This insight enables us to debias graph embedding, i.e., by introducing a bias into the null model, we can in turn remove the bias in the residual, from which the embedding is constructed.

The advantage of our approach is that the bias model is clear and explicit. All debiasing methods assume some null models because they must make judgment about when two nodes should be closer to each other than others. However, the assumptions are often unclear, even when the algorithm itself is simple. By making clear the bias model, we gain greater control over the process of debiasing and a deeper understanding of its consequences.

2.5. Models of network bias

The bias model should exhibit the same type and extent of bias while having no other structure. We construct such a bias model based on the maximum entropy principle (Cimini et al., 2019; Bardoscia et al., 2021). Specifically, we want the noise distribution $P_0(i,j)$ to be maximally random, i.e., maximizing the entropy

$$-\sum_{i,j} P_0(i,j)\log P_0(i,j), \tag{7}$$

while preserving the assortativity over the protected groups:

$$\sum_{i\in\mathcal{C}_\ell}\sum_{i\in\mathcal{C}_k} P_0(i,j) = \sum_{i\in\mathcal{C}_\ell}\sum_{i\in\mathcal{C}_k} P(i,j), \tag{8}$$

for all $1 \leq k,\ell \leq L$, where L is the number of unique protected groups, and $\mathcal{C}_k$ is the set of nodes in the kth group. Finding the $P_0(i,j)$ is a constrained optimization problem, which can be formulated as a Lagrangian:

$$\begin{aligned} \mathcal{L} : = &-\sum_{i,j} P_0(i,j)\log P_0(i,j) \\ &+ \sum_{k=1}^{L}\sum_{\ell=k}^{L}\beta_{k\ell}\left(\sum_{i\in\mathcal{C}_\ell}\sum_{i\in\mathcal{C}_k} P_0(i,j) - \sum_{i\in\mathcal{C}_\ell}\sum_{i\in\mathcal{C}_k} P(i,j)\right). \end{aligned} \tag{9}$$

By taking a functional derivative with respect to P_0 and solving $\partial\mathcal{L}/\partial P_0$ = 0, we obtain

$$P_0(i,j) = \text{Poisson}(\lambda_{ij}), \tag{10}$$

$$\lambda_{ij} = \frac{1}{|\mathcal{C}_{c_i}||\mathcal{C}_{c_j}|}\sum_{i\in\mathcal{C}_\ell}\sum_{j\in\mathcal{C}_k} P(i,j). \tag{11}$$

This bias model is maximally random while preserving the same type and extent of bias in the given network. One can have additional constraints to model a more complex network bias, i.e., degree heterogeneity, degree-degree correlation, and edge directionality. See (Cimini et al., 2019; Bardoscia et al., 2021) for other network models based on the maximum entropy principle.

3. Results

3.1. Datasets

We test GNNs with four different networks (Table1): the political books (?), the political blogs (Adamic & Glance, 2005), the airport network (Ope), twitch gamers (Rozemberczki & Sarkar, 2021) and Facebook network (Traud et al., 2011). These networks include categorical attributes for nodes that we do not want to use to make link prediction (i.e., protected attributes). For all networks, we ignore the edge directionality.

The political blog network represents the political blogs about the 2004 US presidential election connected by hyperlinks. We use the political affiliation assigned to the blogs as the protected attribute.

The airport network is the network of airports connected by edges indicating direct commercial flight. We use the geographical region of the airports as the protected attribute.

Twitch dataset is a social network of Twitch gamers which was collected using public Twitch API. Graph represents a single strongly connected component and edges are the mutual follower relationships between them. We have few features as the metadata for nodes, but for this paper we used language as the protected attribute.

Facebook dataset used here is a friendship network of 100 american colleges and universities. We use gender as a

Table 1. Statistics of network data for link prediction. Information modularity indicates the strength of the communities in a network, where each community is a set of nodes with the same protected attribute. See **??SK: add the robustness modularity paper**.

Network	Nodes	Edges	Protected category (# of categories)	Information Modularity
Political Books	105	441	Political leaning (3)	0.061
Political Blogs	1224	16,715	Political leaning (3)	0.0925
Airport	2898	15,564	Geographical regions (5)	0.112
Twitch	168,114	6,797,557	Language (20)	0.05
Facebook	41,554	1,362,229	Gender (3)	−0.004

protected attribute. Nodes for which gender is unknown are given a separate group.

3.2. Graph neural network models

We test three graph neural networks with different architectures: word2vec (a neural network with one hidden linear layer and the softmax output layer) (Mikolov et al.), GCN (Kipf & Welling, 2017), and GAT (**?**). The GCN and GNN needs node features as input, for which we use $K = 128$ dimensional embedding vectors generated by DeepWalk. We find that using node2vec instead of DeepWalk yield qualitatively the same results (Appendix). To see the effect of debiasing, we compare each model trained with the proposed bias sampling of negative examples with the one with with a conventional uniform sampling. We also employ two families of debiasing methods as baselines. One family is based on the manipulating network data to balance the training data for DeepWalk, i.e., FairWalk (Rahman et al.) and CrossWalk (Khajehnejad et al.). The other family is based on the manipulation of the output embedding, for which we use a debiasing method based on linear projection (Bolukbasi et al., 2016). For a stable baseline we use the method proposed in (Bolukbasi et al., 2016). To apply this method to a graph, we use PCA to determine the bias direction using 20% of the nodes nearest to group centroids. As opposed to the previous work we do not have any gender neutral nodes. We use all the nodes as equalize nodes. We change embedding of each node as

$$\overrightarrow{W} := \overrightarrow{W} - \overrightarrow{D} \cdot proj_{\overrightarrow{W}} \overrightarrow{D} \tag{12}$$

where $\overrightarrow{D}$ is the bias direction.

3.3. Bias in graph embedding

We quantify the biases in graph embedding by the extent to which the sensitive attribute shapes the structure of the embedding. If we were to have an well-debiased graph embedding, the sensitive attributes should not be the main dimensions of the graph embedding. Following the previous study (Bolukbasi et al., 2016), we identify the embedding dimensions that are the most coherent to sensitive attributes and then quantify the ratio of explained variance to total variance of the nodes in the embedding as the level of bias (Fig. **??**). We find that ...

While the sensitive attributes may not entirely shape the embedding structure, they could still have a local impact on the proximity of nodes. To examine the impact on the local proximity, we test to what extent each node is equidistance to the protected groups. More specifically, for each node i with embedding vector u_i, we compute the cosine similarity (i.e., $\cos(u_i, u_\ell)$) to the centroid vector u_ℓ of each group $c = \ell$. Then, we compute the *local disparity* D_i by the variance over all groups, i.e.,

$$D_i := \frac{1}{L}\sum_{\ell=1}^{L}\left(\cos(u_i, u_\ell) - \frac{1}{L}\sum_{\ell'=1}^{L}\cos(u_i, u_{\ell'})\right)^2. \tag{13}$$

We find that the local disparity drops for majority of nodes in the network when trained using biased negative sampling (Fig. **??**). The reduction of disparity is overall consistent across different model architecture and networks.

In summary, our training framework consistently brings a reduction of biases in graph neural networks compred to their unbiased counterparts across different networks both at global and local levels. The reduction of bias is comparable or substantial compared to the baseline debiasing methods. These results demonstrate that our training framework for debiasing is consistently effective and model-independent in reducing the biases in graph embedding.

3.4. Prediction performance

Debiasing an embedding involves masking certain information within the embedding, which can potentially lead to a significant reduction in the embedding's utility. Indeed, the debiasing methods including our method and the baselines all results in the decreased link prediction performance. However, the amount of the decrease differs method by methods. In fact, our training method results in the least decreases in the performance among all the debiasing methods (Fig. xxx).

We also examine the utility of embedding at a local node. For each node, we find the 50 closest neighbors in the given embedding. Then, we calculate the average precision of the node similarity for the test edges. xxxx

4. Conclusion

We proposed a novel training framework to jointly improve fairness and quality of graph embeddings. We validated the proposed framework by training three models ranging from a shallow to deep neural networks. Notably, our approach makes the biased structural features inconseqential to model training, rather than heuristically removing biased input data or biased dimensions in the generated embedding. Overall, our proposed framework presents a new way to address the issue of biases in graph data, and can be applied to a variety of downstream AI applications.

There are certain limitations to be acknowledged. First, our approach requires a biased model encoding biased associations. We used the degree-corrected stochastic block model as the biased model by following the previous study (Kojaku et al.). Learning a suitable biased model for debiasing warrants future work. Second, our approach assumes that the sensitive attribute is known and labeled, which may not always be the case in real-world scenarios. Third, while we have focused on direct associations between sensitive attributes, our approach may not completely eliminate bias due to indirect associations of biased attributes through other attributes (Gonen & Goldberg, 2019). Despite these limitations, our study demonstrates that by biasing the sampling of negative examples, it is possible to reduce bias while maintaining high accuracy in graph embedding.

References

Gender bias on wikipedia. URL https://en.wikipedia.org/w/index.php?title=Gender_bias_on_Wikipedia&oldid=1128262996. Page Version ID: 1128262996.

Google fixes translate tool after accusations of sexism. URL https://www.independent.co.uk/life-style/women/google-translate-sexist-masculine-feminine-he-said-she-said-english-spanish-languages-a8672 html. Section: Lifestyle.

Openflights: Airport and airline data. https://openflights.org/data.html. (Accessed on 01/12/2023).

Adamic, L. A. and Glance, N. The Political Blogosphere and the 2004 U.S. Election: Divided They Blog . In *Proceedings of the 3rd International Workshop on Link Discovery*, LinkKDD '05, pp. 36–43, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595932151. doi: 10.1145/1134271.1134277. URL https://doi.org/10.1145/1134271.1134277.

Bardoscia, M., Barucca, P., Battiston, S., Caccioli, F., Cimini, G., Garlaschelli, D., Saracco, F., Squartini, T., and Caldarelli, G. The physics of financial networks. *Nature Reviews Physics*, 3(7):490–507, 2021.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Bolukbasi, T., Chang, K., Zou, J. Y., Saligrama, V., and Kalai, A. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *CoRR*, abs/1607.06520, 2016. URL http://arxiv.org/abs/1607.06520.

Bose, A. and Hamilton, W. Compositional fairness constraints for graph embeddings. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 715–724. PMLR. URL https://proceedings.mlr.press/v97/bose19a.html. ISSN: 2640-3498.

Bourli, S. and Pitoura, E. Bias in knowledge graph embeddings. In *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 6–10. doi: 10.1109/ASONAM49781.2020.9381459. ISSN: 2473-991X.

Cimini, G., Squartini, T., Saracco, F., Garlaschelli, D., Gabrielli, A., and Caldarelli, G. The statistical physics of real-world networks. *Nature Reviews Physics*, 1(1):58–71, 2019.

Derrow-Pinion, A., She, J., Wong, D., Lange, O., Hester, T., Perez, L., Nunkesser, M., Lee, S., Guo, X., Wiltshire, B., Battaglia, P. W., Gupta, V., Li, A., Xu, Z., Sanchez-Gonzalez, A., Li, Y., and Velickovic, P. ETA prediction with graph neural networks in google maps. *CoRR*, abs/2108.11482, 2021. URL https://arxiv.org/abs/2108.11482.

Dyer, C. Notes on noise contrastive estimation and negative sampling, 2014. URL https://arxiv.org/abs/1410.8251.

Edwards, H. and Storkey, A. J. Censoring representations with an adversary. *CoRR*, abs/1511.05897, 2015.

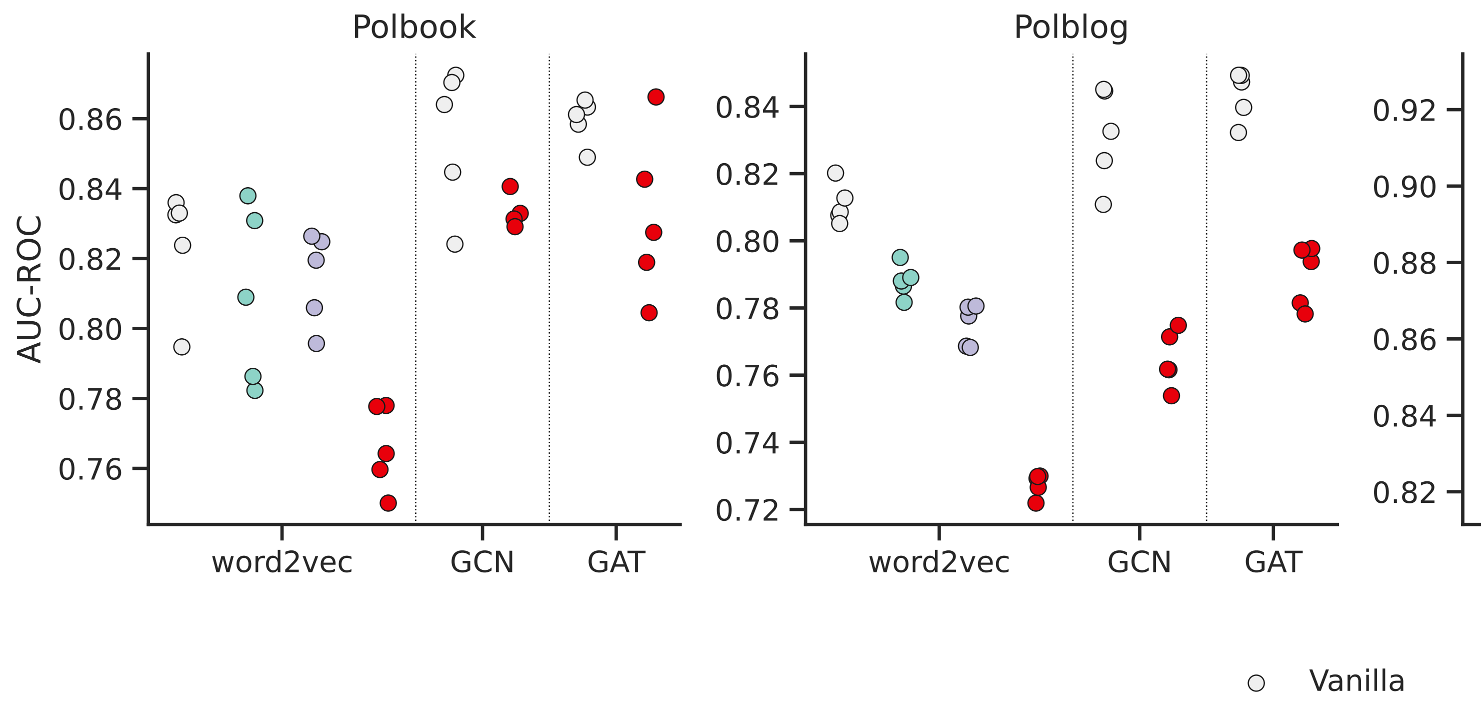


Figure 2. Link Prediction benchmark results for different datasets. Fig **??** (A-E) shows the AUC-ROC scores for which negative edges were generated using **??**. Fig **??** (F-H) shows the disparity scores. Disparity here is defined by how invariant the model is with respect to the protected attribute in predicting the edges for test network.

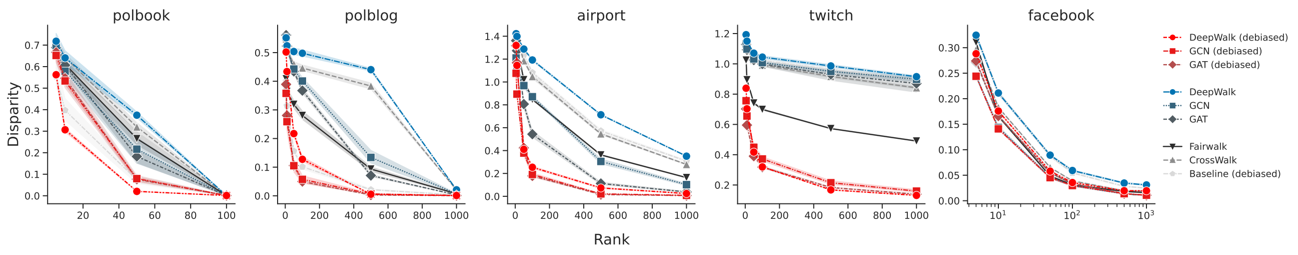


Figure 3. Relative Entropy of predictive group membership distribution and original group membership distribution vs k (number of nearest neighbors)

Gonen, H. and Goldberg, Y. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. *CoRR*, abs/1903.03862, 2019. URL http://arxiv.org/abs/1903.03862.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 855–864. Association for Computing Machinery, July 2016. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939754. URL https://doi.org/10.1145/2939672.2939754. arXiv:1607.00653 [cs, stat].

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Teh, Y. W. and Titterington, M. (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html. ISSN: 1938-7228.

Hamilton, W. L., Ying, R., and Leskovec, J. Induc-

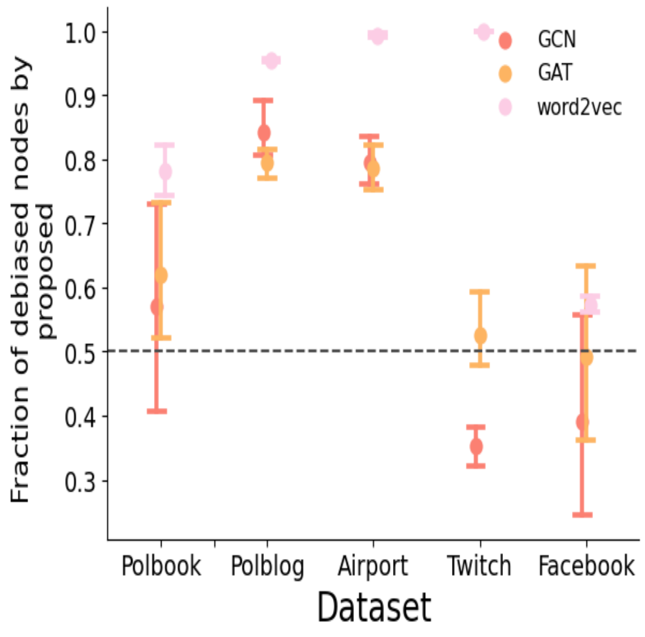


Figure 4. Disparity per Node, this plot demonstrates ratio of nodes for which the standard deviation of cosine similarity from the centroid of all protected groups decreased in proposed versus compared to baseline models. We see a noticeable drop in the cumulative disparity of the set of nodes. Baselines are the models trained with randomly Sampled negative nodes while proposed models are trained using proposed biased negative sampling. **at: Currently plotting just one run, should we plot average?**

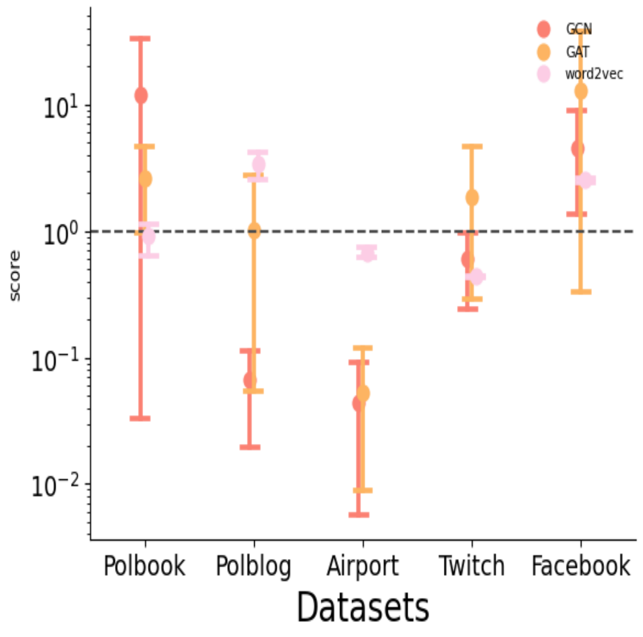


Figure 5. Global Fairness

tive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017. URL http://arxiv.org/abs/1706.02216.

Jumper, J. M., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Zídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D. A., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021.

Khajehnejad, A., Khajehnejad, M., Babaei, M., Gummadi, K. P., Weller, A., and Mirzasoleiman, B. CrossWalk: Fairness-enhanced node representation learning. URL http://arxiv.org/abs/2105.02725.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks, 2017.

Kojaku, S., Yoon, J., Constantino, I., and Ahn, Y.-Y. Residual2vec: Debiasing graph embedding with random graphs. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24150–24163. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2021/hash/ca9541826e97c4530b07dda2eba0e013-Abstract.html.

Kunegis, J. KONECT – The Koblenz Network Collection. In *Proc. Int. Conf. on World Wide Web Companion*, pp. 1343–1350, 2013. URL http://dl.acm.org/citation.cfm?id=2488173.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html.

Mirhoseini, A., Goldie, A., Yazgan, M., Jiang, J. W., Songhori, E. M., Wang, S., Lee, Y.-J., Johnson, E., Pathak, O., Nazi, A., Pak, J., Tong, A., Srinivasa, K., Hang, W., Tuncer, E., Le, Q. V., Laudon, J., Ho, R., Carpenter, R., and Dean, J. A graph placement methodology for fast chip design. *Nature*, 594 7862:207–212, 2021.

Newman, M. *Networks*. Oxford University Press, 2nd edition edition. ISBN 978-0-19-880509-0.

Pages, T. S. Nikon camera says asians: People are always blinking - sociological images. URL https://thesocietypages.org/socimages/2009/05/29/nikon-camera-says-asians-are-always-blinking/.

Perozzi, B., Al-Rfou, R., and Skiena, S. DeepWalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '14, pp. 701–710. Association for Computing Machinery. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623732. URL https://doi.org/10.1145/2623330.2623732.

Rahman, T., Surma, B., Backes, M., and Zhang, Y. Fairwalk: Towards fair graph embedding. pp. 3289–3295. URL https://www.ijcai.org/proceedings/2019/456.

Ravfogel, S., Elazar, Y., Gonen, H., Twiton, M., and Goldberg, Y. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7237–7256, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.647. URL https://aclanthology.org/2020.acl-main.647.

Rozemberczki, B. and Sarkar, R. Twitch gamers: a dataset for evaluating proximity preserving and structural role-based node embeddings, 2021.

Solaiman, I. and Dennison, C. Process for adapting language models to society (PALMS) with values-targeted datasets. In *Advances in Neural Information Processing Systems*, volume 34, pp. 5861–5873. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2021/hash/2e855f9489df0712b4bd8ea9e2848c5a-Abstract.html.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. LINE: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pp. 1067–1077. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-3469-3. doi: 10.1145/2736277.2741093. URL https://doi.org/10.1145/2736277.2741093.

Traud, A. L., Mucha, P. J., and Porter, M. A. Social structure of facebook networks. *CoRR*, abs/1102.2166, 2011. URL http://arxiv.org/abs/1102.2166.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks, 2017. URL https://arxiv.org/abs/1710.10903.

Xing, Y., Song, Q., and Cheng, G. On the algorithmic stability of adversarial training. *Advances in Neural Information Processing Systems*, 34:26523–26535, 2021.

Yang, S. Networks: An introduction by m. e. j. newman. *The Journal of Mathematical Sociology*, 37(4):250–251, 2013. doi: 10.1080/0022250X.2012.744247. URL https://doi.org/10.1080/0022250X.2012.744247.

Yoon, J., Yang, K.-C., Jung, W.-S., and Ahn, Y.-Y. Persona2vec: a flexible multi-role representations learning framework for graphs. *PeerJ Computer Science*, 7: e439, March 2021. ISSN 2376-5992. doi: 10.7717/peerj-cs.439. URL https://doi.org/10.7717/peerj-cs.439.

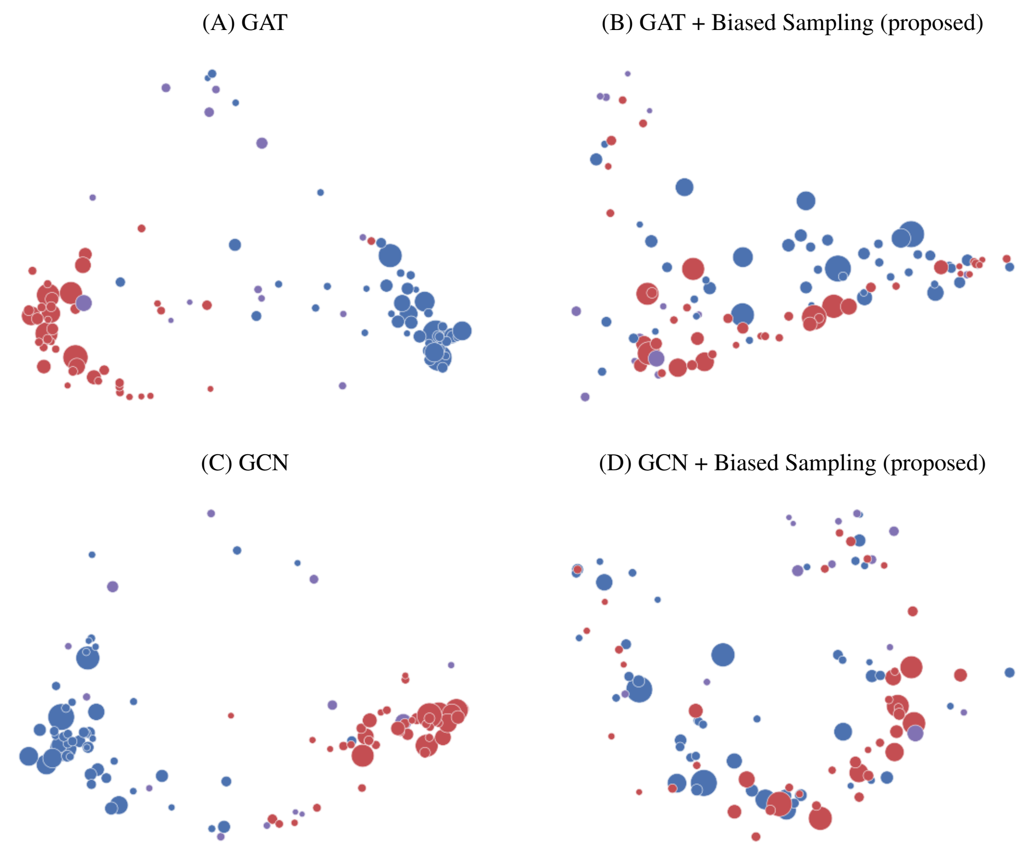


Figure 6. Demonstration on Political Book Dataset. Each node represents a book. The color of the node indicates the political alignment of the book, blue for liberal, red for conservative, and purple for neutral. The size of each node is proportional to the degree of a node in the original network. Images are generated using PCA on 16 dimensional embeddings. In each of these case nodes are represented by their 16 dimensional node2vec features

A. Appendix

A.1. node2vec with FairGraph

Apart from trying our method with DeepWalk, we also tried it with node2vec. In this section we demonstrate that both of our contributions, FairGraph and Biased NCE Sampling work with any model and feature generation method.

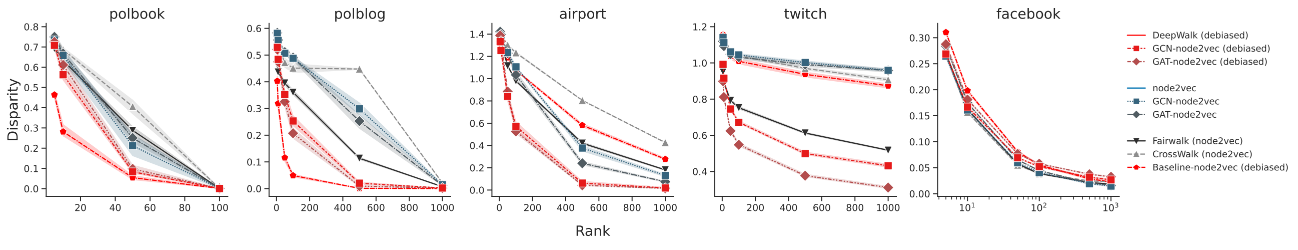
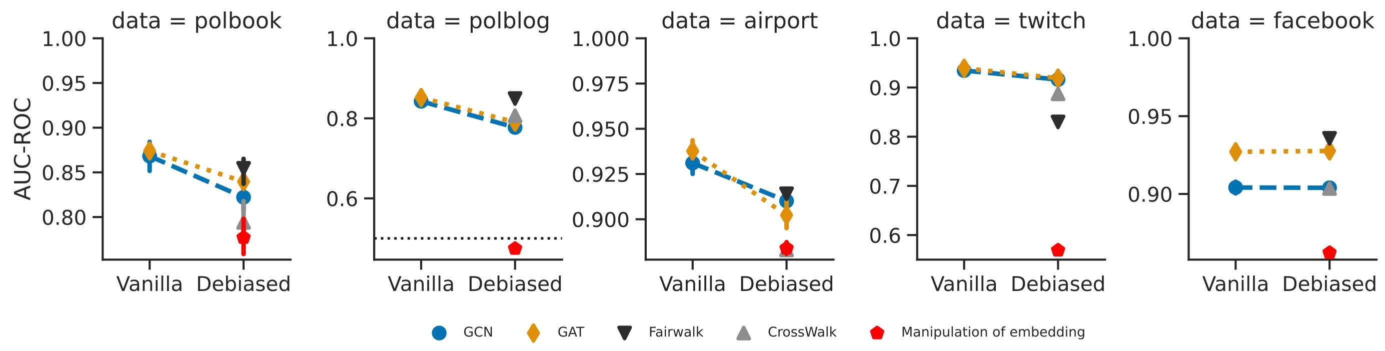


Figure 7. Disparity with node2vec features

A demonstration of the proposed method on node2vec features is shown in Fig.6. In this plot we show the stark difference between baseline and proposed method on two GNN architectures, GAT and GCN (two rows). Here we have included PCA We can see that the proposed method can reduce the separability of the groups.

[t]



[t]

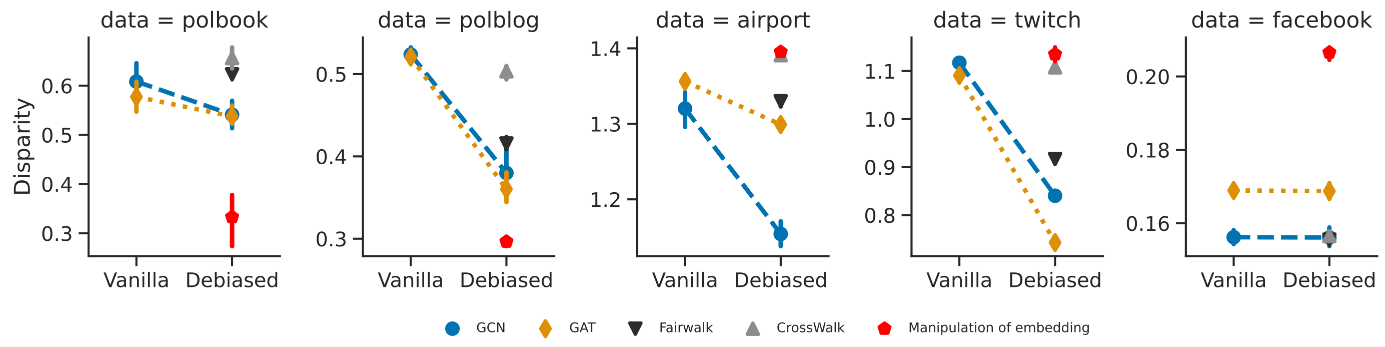


Figure 8. Link prediction (node2vec features)

A.2. Training Hyperparameters

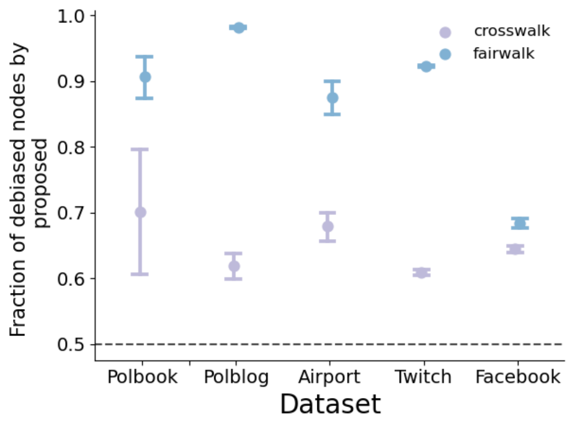
All graph neural networks take 128 dimensional node features generated by DeepWalk/node2vec as input. Size of the neural network is different for different datasets. For smaller networks (Airport, Political Blogs and Political Book) we use 3 layers for GCNConv or GATConv stacked over one another, for Facebook 4 and for Twitch we use 5 followed by 2 linear layers. For different experiments, these DeepWalk/node2vec features are generated independently from different training networks. In cases of CrossWalk and Fairwalk, the training network is first modified using these algorithms and then a node2vec or DeepWalk model is trained to output a result of 128 dimensional embeddings. Thus result of all the models (GNNs, CrossWalk, Fairwalk, DeepWalk/Fairwalk as baseline) are embeddings of 128 dimensions for each node.

For Fairwalk, we train a random walk sampler for 1 epoch with a window length of 10 and walk length of 40. In case of CrossWalk there are two important hyperparameters, α and p which are used to modify the training network. For α we use a value of 0.7 while for p it is 2. α is the multiplicating factor in the reweighting process. It controls the probability of a boundary node connected to other groups. This value is proportional to the number of inter-group connections in the resulting network. p is the exponent in the reweighting process. It controls the amount of information that propagates from one group to other groups. The higher the value of p, the more is the information flow. For DeepWalk and node2vec we use 1 epoch of training with a window length of 10.

For GNNs we use either 3, 4 or 5 layers of GATConv or GCNConv (depending on the dataset) stacked over one another. Output of each of those hidden layers is 64 dimensional. The output of the model is a 128 dimensional embedding of each node. For GATs we use 2 heads in each GATConv layer. A dropout of 0.2 is performed on the output of these layers before these are fed into a sequential module of two linear layers output of which is the embedding of the node.

These models are trained using Adam optimizer with a learning rate of 0.001 ($\frac{0.001}{32}$ for Twitch) and early stopping is used to prevent overfitting. Early stopping is configured to stop training when more than half of the batches reach a loss of zero. The model is trained for 300 epochs for Airport, 600 epochs for Political Book and Political Blogs datasets, 30 for Facebook and 20 for Twitch. The model is trained on a GPU with a batch size of 256 (8 for Twitch). Tesla A 100s were used to train all the models, with PyTorch being the machine learning tool of choice.

(a) Local Fairness



(b) Global Fairness

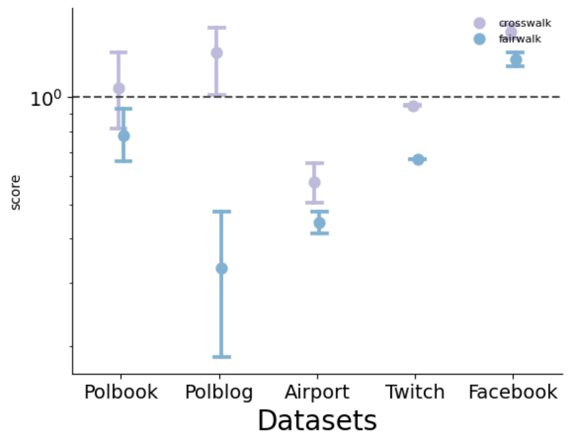


Figure 9. Comparison of Crosswalk and Fairwalk with vanilla deepwalk