2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Debiasing Graph Representations via Metadata-Orthogonal Training

John Palowitch
Google Research
palowitch@google.com

Bryan Perozzi
Google Research
bperozzi@acm.org

***Abstract*—In real world graphs, the formation of edges can be associated with certain sensitive features of the nodes (e.g. gender, community, reputation). In this paper we argue that when such associations exist, any downstream Graph Neural Network (GNN) will be implicitly biased by these structural correlations. To allow control over this phenomenon, we introduce the Metadata-Orthogonal Node Embedding Training (MONET) unit, a general neural network module for performing training-time linear debiasing of graph embeddings. MONET operates by ensuring that the node embeddings are trained on a hyperplane orthogonal to that of the node features (metadata). Unlike debiasing approaches in similar domains, our method offers exact guarantees about the correlation between the resulting embeddings and any sensitive metadata. We illustrate the effectiveness of MONET though our experiments on a variety of real world graphs against challenging baselines (e.g. adversarial debiasing), showing superior performance in tasks such as preventing the leakage of political party affiliation in a blog network, and preventing the gaming of embedding-based recommendation systems.**

I. Introduction

Graph embeddings – continuous, low-dimensional vector representations of nodes – have been eminently useful in network visualization, node classification, link prediction, and many other graph learning tasks [1, 2]. While graph embeddings can be learned directly by unsupervised algorithms using relational edges, [e.g. 3, 4, 5, 6, 7], there is often non-relational information available for each node in the graph. This information, frequently called node *attributes* or node *metadata*, can contain information that is useful for prediction tasks including demographic, geo-spatial, or textual features.

The interplay between a node's metadata and edges is an active area of research. Interestingly, in many applications, metadata can be measurably related to a graph's structure [8, 9]. As such, metadata can enhance graph learning models [10, 11], and conversely, graphs can be used as regularizers in (semi-)supervised models of node features [12, 13]. Furthermore, metadata are commonly used as evaluation data for graph embeddings [14]. For example, node embeddings trained on a Flickr user graph were shown to predict user-specified Flickr "interests" [3]. This is plausibly because users in the Flickr graph tend to follow users with similar interests, illustrating an association between node topology and node metadata.

However, despite the usefulness and prevalence of metadata in graph learning, there are instances where it is desirable to design a system to *avoid* the effects of a particular kind of *sensitive* data. For instance, practitioners using pre-trained

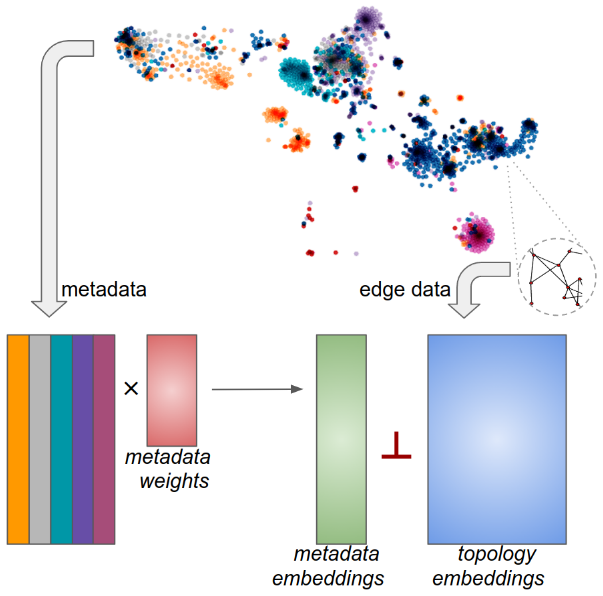


Figure 1: Illustration of MONET. A 2-D graph projection, colored by node attributes, illustrates clear topology-metadata association. MONET concurrently embeds the metadata, then debiases topology embeddings from the metadata embeddings.

graph embeddings in a predictive model may wish to make the embeddings uncorrelated with (say) demographic information.

At first glance, this may seem like an artificial problem – surely one could just avoid the problem by training graph embeddings without metadata information. However, that approach does not control for inherent correlations that may exist between the sensitive metadata and the edge data. In other words, if the edges of the graph are correlated with sensitive metadata, then any algorithm which does not explicitly model and remove this correlation will be biased as a result of it.

To this end, we propose two desiderata for handling sensitive metadata in Graph Neural Networks (GNNs):

D1. The influence of metadata on the graph topology is modeled in a partitioned subset of embedding space, providing interpretability to the overall graph representation, and the option to remove metadata dimensions.

D2. Non-metadata or "topology" embeddings are debiased from metadata embeddings with a *provable guarantee* on the level of remaining bias.

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

In this work we propose a novel GNN technique that satisfies both of these desiderata. Our method, the Metadata-Orthogonal Node Embedding Training (MONET) unit, operates by ensuring that metadata embeddings are trained on a hyperplane orthgonal to that of the topology embeddings (as conceptually illustrated in Figure 1). Specifically, our contributions are as follows:

1) The Metadata-Orthogonal Node Embedding Training (MONET) unit, a novel GNN algorithm which jointly embeds graph topology and graph metadata while enforcing linear decorrelation between the embedding spaces.
2) Analysis which proves that addressing desiderata D1 alone – partitioning metadata embeddings – still produces biased topology embeddings, and that MONET corrects this.
3) Experimental results on real world graphs which show that MONET can successfully debias topology embeddings while relegating metadata information to separate dimensions – achieving superior results to state-of-the-art adversarial and NLP-based debiasing methods.

II. Preliminaries

Early graph embedding methods involved dimensionality reduction techniques like multidimensional scaling and singular value decomposition [14]. In this paper we use graph neural networks trained on random walks, similar to DeepWalk [3]. DeepWalk and many subsequent methods first generate a sequence of random walks from the graph, to create a "corpus" of node "sentences" which are then modeled via word embedding techniques (e.g. word2vec [15] or GloVe [16]) to learn low dimensional representations that preserve the observed co-occurrence similarity.

Let W be a d-dimensional graph *embedding* matrix, $W \in R^{n\times d}$, which aims to preserve the low-dimensional structure of a graph ($d << n$). Rows of W correspond to nodes, and node pairs i, j with large dot-products $W_i^T W_j$ should be structurally or topologically close in the graph. In this paper we illustrate graph embedding debiasing using a recently proposed GloVe-based graph embedding model [16, 17]:

$$L_{\text{GloVe}} = \sum_{i,j\leq n} f_\alpha(C_{ij})(a_i + b_j + U_i^T V_j - \log(C_{ij}))^2. \quad (1)$$

Above, $U, V \in R^{n\times d}$ are the "center" and "context" embeddings, $a, b \in R^{n\times 1}$ are the biases, C is the walk-distance-weighted context co-occurrences, and f_α is the loss smoothing function [16]. We use the GloVe model in the next section, and throughout the paper, to illustrate the incorporation of metadata embeddings and the MONET unit. However, these innovations are generally deployable modules. To illustrate this, we also describe a MONET unit for DeepWalk [3] in Section III.

Notation. In this paper, given a matrix $A \in R^{n\times d}$ and an index $i \in 1, \ldots, n$, A_i denotes the $d \times 1$ i-th row vector of A. Column indices will not be used. $\mathbf{0}_{n\times d}$ denotes the $n \times d$ zero matrix, and $||\cdot||_F$ denotes the Frobenius norm.

A. Related Work

The combination of desiderata D1 and D2 introduced in Section 1 formulates a bias in machine learning (ML) problem for graphs that goes unaddressed by the vast majority of the graph embeddings literature. Existing embedding methods for graphs with metadata [e.g. 18, 10, 11, 19, 20, 21] do not consider D2, and most do not not fully address D1 either. In the GNN literature metadata are treated as features in predictive models, rather than attributes to be controlled [1].

There are two methods predating this work, [22] and FairWalk [23], that address bias in graph learning. [22] is a method built for supervised link-prediction in recommender systems that uses adversarial training against sensitive attributes, and does not directly apply to the unsupervised embedding problems we discuss here. However, we have adapted the method to our domain and presented it as one of our unsupervised baselines for comparison. FairWalk is an approach for generating random walks such that if a walk visits node u, all attribute categories among the neighbors of u are equi-probable when choosing the next node. As we will show in our experiments, the FairWalk procedure can fail to capture complex correlation structure between metadata and edge formation.

Definitions of bias. The bias objective we focus on in this paper (focus of desiderata D2) is the linear correlation between the debiased embeddings and sensitive metadata, defined explicitly in Section 3. This objective relates to the ability of embeddings to predict the metadata, an ability which has been used as an empirical measure of model bias in recent fairness literature [24, 22]. Similar linear objectives are standard in debiasing literature [25, 26, 24] including the parallel work [27]. Other objectives involve various definitions of statistical dependence [28], which have recently been addressed with adversarial learning [29], including in a method specifically for graph recommender systems [22], which is most similar to our work. Some of these independence-oriented objectives could arguably mitigate non-linear bias relationships that our approach would ignore. However, adversarial approaches in this space do not *guarantee* debiasing unless the adversary tuning parameter is too large for practical purposes, as pointed out in [22]. We show in this paper that the approach taken by [22] results in more bias than our method when used in both linear and nonlinear classifiers.

III. Metadata Embeddings and Orthogonal Training

In this section we present the components of MONET. First, in Section III-A, we extend traditional graph representation to include metadata embeddings, achieving desiderata D1. Next, in Section III-B, we prove that a model with just D1 will still leak metadata information to the topology embeddings. Then, in Section III-C we present MONET's key algorithm for training metadata and topology embeddings orthogonally, achieving desiderata D2. Finally, we conclude with some analysis of MONET in Section 3.4.

A. Jointly Modeling Metadata & Topology

A natural first approach to controlling metadata effects is to introduce a partition of the embedding space that models them explicitly, adding interpretability and separability to the overall

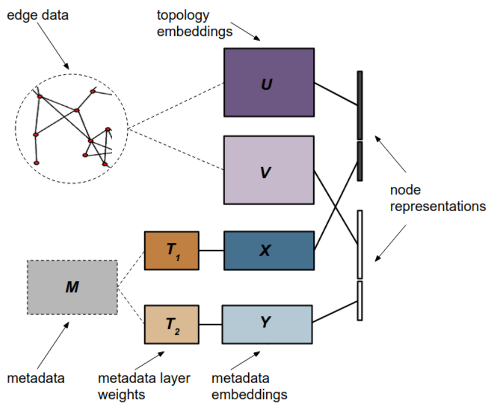


Figure 2: Illustration of GloVe$_{\text{meta}}$. U and V are topology embedding layers. A single-layer feed-forward neural network creates metadata embeddings layer X and Y. The concatenations $[U, X]$ and $[V, Y]$ create the output node representations.

representation. We denote the node metadata matrix as $M \in R^{n\times m}$, where the metadata for node i is contained in the row vector M_i. To achieve D1, we feed M through a single-layer neural network with weights T. Figure 2 shows a realization of this idea, using the GloVe embedding model, which we denote GloVe$_{\text{meta}}$. Here, two weight matrices T_1, T_2 are needed to embed the metadata for both center and context nodes. This network yields metadata *embeddings* X, Y, which correspond to the standard topology embeddings U, V (respectively), and the cross-pairs of embeddings are concatenated to produce a joint representation. The resulting modified loss is:

$$L_{\text{GloVe}_{\text{meta}}} = \sum_{i,j\le n} f_\alpha(C_{ij})(a_i+b_j+U_i^TV_j+X_i^TY_j-\log(C_{ij}))^2. \tag{2}$$

While we use GloVe in this paper's experiments, metadata embeddings can be incorporated in any GNN which utilizes a node-wise graph representation. For instance, the well-known DeepWalk [3] loss, which is based on word2vec [15], would incorporate metadata embeddings as follows:

$$-\sum_{i,j\in\mathcal{W}} \log(U_i^TV_j + X_i^TY_j) - \sum_{k\in K_i} \log(-U_i^TV_k - X_i^TY_k).$$

Above, $\mathcal{W}$ is the set of context pairs from random walks, and K_i is a set of negative samples associated with node i.

This new modeling approach, which augments standard graph representations with a metadata partition, provides users of graph embeddings with the option to include or exclude metadata-learned dimensions. Furthermore, suppose that the metadata (e.g. demographic information) are indeed associated with the formation of links in the graph. In this case, ostensibly, the dedicated metadata embedding space could relieve the topology dimensions of the responsibility to encode this relationship, thereby reducing metadata bias in those dimensions. In our empirical study (Section 4) we show that to some extent, this does actually occur. However, we find – empirically and theoretically – that this naïve approach does not guarantee completely decorrelated topology and metadata embeddings. In fact, suprisingly, we find that most of the metadata bias sufferred by standard baselines remain in the topology dimensions. This is a phenomenon we call *metadata leakage*, which we formalize and prove in the next section.

B. Metadata Leakage in GNNs

Here, we formally define *metadata leakage* for general topology and metadata embeddings, and show how it can occur even in embedding models with partitioned metadata embeddings. This motivates the need for both D1 and D2 in a complete approach to controlling metadata effects in GNNs.

Definition 1. *The* ***metadata leakage*** *of metadata embeddings $Z \in R^{n\times d_Z}$ into topology embeddings $W \in R^{n\times d}$ is defined $\mathcal{ML}(Z, W) := ||Z^TW||_F^2$. We say that there is no metadata leakage if and only if $\mathcal{ML}(Z, W) = 0$.*

Without a more nuanced approach, metadata leakage can occur even in embedding models with a metadata partition, like those discussed in Section III-A. To demonstrate this we consider a reduced metadata-laden GloVe loss with unique topology and metadata representations W and $Z = MT$:

$$L^*_{\text{GloVe}_{\text{meta}}} = \sum_{i,j\le n} f_\alpha(C_{ij})(a_i+a_j+W_i^TW_j+Z_i^TZ_j-\log(C_{ij}))^2 \tag{3}$$

We now show that under a random update of GloVe under Eq. (3), the expected metadata leakage is non-zero. Specifically, let (i, j) be a node pair from the co-occcurrences C, and define $\delta_W(i, j)$ as the incurred Stochastic Gradient Descent update $W' \leftarrow W + \delta_W(i, j)$. Suppose there is a "ground-truth" metadata transformation $B \in R^{m\times d_B}$, and define ground-truth metadata embeddings $\tilde{Z} := MB$, which represent the "true" dimensions of the metadata effect on C. Define $\Sigma_B := BB^T$ and $\Sigma_T := TT^T$. With expectations taken with respect to the sampling of a pair (i, j) for Stochastic Gradient Descent, define $\mu_W := E[W_i]$ and $\Sigma_W := E[W_iW_i^T]$. Define μ_M, Σ_M similarly. Then our main Theorem is as follows:

Theorem 1. *Assume $\Sigma_W = \sigma_W I_d$ for $\sigma_W > 0$, $\mu_W = \mathbf{0}_{d\times1}$, and $\mu_M = \mathbf{0}_{m\times1}$. Suppose for some fixed $\theta \in R$ we have $\log(C_{ij}) = \theta + \tilde{Z}_i^T\tilde{Z}_j$. Let (i, j) be a randomly sampled co-occurrence pair and W' the incurred update. Then if $E[M_iW_i^T] = \beta \in R^{m\times d}$, we have*

$$E[\mathcal{ML}(Z, W')] \ge 2||T^T\left[\Sigma_M(\Sigma_B - \Sigma_T) + (n - \sigma_W)I_m\right]\beta||_F^2.$$

Proof. From derivatives of L^* the i-th row of $\delta_W(i, j)$ is:

$$(\theta + \tilde{Z}_i^T\tilde{Z}_j - Z_i^TZ_j - W_i^TW_j - a_i - a_j)W_j^T \tag{4}$$

Taking expected values and applying independence, we have:

$$E[Z_i\delta_W(i, j)_i^T] = T^T\left[\Sigma_M(\Sigma_B - \Sigma_T) - \sigma_W I_d\right]\beta. \tag{5}$$

Note $E[Z^T\delta_W(i, j)] = E[Z_i\delta_W(i, j)_i^T] + E[Z_j\delta_W(i, j)_j^T]$ and $E[M_iW_i^T] = \beta \Rightarrow M^TW = n\beta \Rightarrow Z^TW = nT^T\beta$, thus

$$E[Z^TW'] = 2T^T\left[\Sigma_M(\Sigma_B - \Sigma_T) + (n - \sigma_W)I_m\right]\beta. \tag{6}$$

Applying Jensen's Inequality completes the proof. □

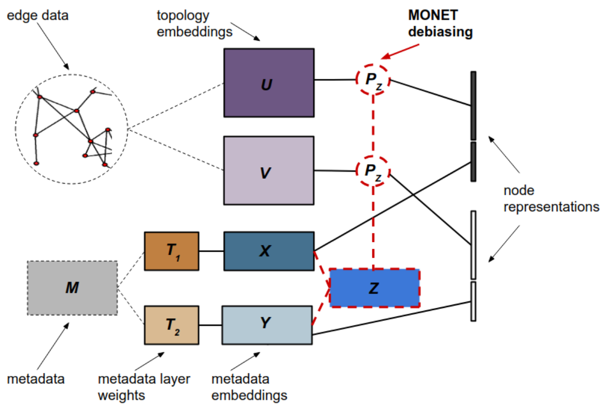


Figure 3: MONET implemented in GloVe. The MONET unit adds to GloVe$_{\text{meta}}$ by using the metadata embedding $Z = X+Y$ to debias U and V via the orthogonal projection P_Z.

Algorithm 1: MONET Unit Training Step

```
Input: topology embedding $W$, metadata embedding $Z$
  procedure FORWARD PASS DEBIASING($W$, $Z$)
      Compute $Z$ left-singular vectors $Q_Z$ and projection:
        $P_Z \leftarrow I_{n\times n} - Q_Z Q_Z^T$
      Compute orthogonal topology embedding:
        $W^\perp \leftarrow P_Z W$
      return debiased graph representation $[W^\perp, Z]$
  end procedure
  procedure BACKWARD PASS DEBIASING($\delta_W$)
      Compute orthogonal topology embedding update:
        $\delta_W^\perp \leftarrow P_Z \delta_W$
      Apply update:
        $W^\perp \leftarrow W^\perp + \delta_W^\perp$
      return debiased topology embedding $W^\perp$
  end procedure
```

Importantly, Σ_T and σ_W are neural network hyperparameters, so we give a useful Corollary:

Corollary 1. *Under the assumptions of Theorem 1, $E[\mathcal{ML}(Z, W)] = \Omega(n)$ as $n \to \infty$.*

Note that under reasonable GNN initialization schemes, T and β are random perturbations. Thus, Corollary 1 implies the surprising result a metadata embedding partition is not sufficient to prevent metadata leakage in practical settings.

C. MONET: Metadata-Orthogonal Node Embedding Training

Here, we introduce the Metadata-Orthogonal Node Embedding Training (MONET) unit for training joint topology-metadata graph representations $[W, Z]$ without metadata leakage. MONET explicitly prevents the correlation between topology and metadata, by using the Singular Value Decomposition (SVD) of Z to *orthogonalize* updates to W during training.

MONET. The MONET unit (Algorithm 1) is a two-step algorithm applied to the training of a topology embedding in a neural network. The input to a MONET unit is a metadata embedding $Z \in R^{n\times d_z}$ and a target topology embedding $W \in R^{n\times d}$ for debiasing. Let $Q_Z \in R^{n\times d_{z0}} (d_{z0} \le d_z)$ be the left-singular vectors of Z, and define $P_Z := I_{n\times n} - Q_Z Q_Z^T$. In the forward pass, debiased topology weights are obtained by using the projection $W^\perp = P_Z W$. Similarly, $W^\perp$ is used in place of W in subsequent GNN layers. In the backward pass, MONET also debiases the backpropagation update to the topology embedding, δ_W, using $\delta_W^\perp = P_Z \delta_W$.

Straightforward properties of the SVD show that the MONET unit, defined in Algorithm 1, eliminates metadata leakage:

Theorem 2. $\mathcal{ML}(Z, W^\perp) = \mathcal{ML}(Z, \delta^\perp) = 0$.

We note that in this paper we have only considered *linear* metadata leakage; provable guarantees for debiasing nonlinear topology/metadata associations is an area of future work.

Implementation (MONET$_G$). We demonstrate MONET in our experiments by applying Algorithm 1 to Eq. (2), which we denote as MONET$_G$. We orthogonalize the input and output topology embeddings U, V with the summed metadata embeddings $Z := X + Y$. By linearity, this implies Z-orthogonal training of the summed topology representation $W = U + V$. We note that working with the sums of center and context embeddings is the standard way to combine these matrices [16]. Figure 3 gives a full illustration of MONET$_G$.

Relaxation. A natural generalization of the MONET unit is to parameterize the level of orthogonalization at each training step. Specifically, we introduce a parameter $\lambda \in [0.0, 1.0]$ which controls the extent to which topology embeddings are projected onto the metadata-orthogonal hyperplane:

$$P_Z^{(\lambda)} := I_{n\times n} - \lambda Q_Z Q_Z^T \tag{7}$$

$P_Z^{(\lambda)}$ takes the role of P_Z in 1. Using MONET with $\lambda = 1.0$ removes all linear bias, whereas using $\lambda = 0.0$ prevents any debiasing. In Section 4.2, we explore how λ affects the trade-off between linear debiasing and task accuracy.

D. Analysis

We briefly remark about the algorithmic complexity of MONET, and the interpretation of its parameters.

Algorithmic Complexity. The bottleneck of MONET occurs in the SVD computation and orthogonalization. In our setting, the SVD is $O(nd_z^2)$ [30]. The matrix P_Z need not be computed to perform orthogonalization steps, as $P_Z W = W - Q_Z(Q_Z^T W)$, and the right-hand quantity is $O(ndd_z)$ to compute. Hence the general complexity of the MONET unit is $O(nd_z \max\{d, d_z\})$. The MONET unit is only applied at every batch, so if $d << n$ (expected in typical settings), the complexity is dwarfed by the overall training complexity. In the experiments section we compare the wall clock time of MONET and baselines, showing only about a 23% overall wall time increase from standard GloVe.

MONET can also apply when only a small subset of the metadata is considered sensitive and of interest to debias. Suppose this subset of dimensions is of size $d_s < d_z$ – (e.g. age or political affiliation metadata) while the remaining $d_z - d_s$ dimensions correspond to online comment bag-of-words. In this case, one can compute P_Z only from those d_s metadata dimensions, which greatly reduces the SVD complexity.

Metadata Parameter Interpretation. The i, j terms in the sum of the loss for GloVe models with metadata (GloVe$_{\text{meta}}$

and MONET$_G$) involve the dot product $X_i^T Y_j = M_i^T T_1 T_2^T M_j$. That expansion suggests that the matrix $\Sigma_T := T_1 T_2^T$ contains all pairwise metadata dimension relationships. In other words, Σ_T gives the direction and magnitude of the raw metadata effect on log co-occurrence, and is therefore a way to measure the extent to which the model has captured metadata information. We will refer to this interpretation in the experiments that follow. An important experiment will show that applying the MONET algorithm increases the magnitude of Σ_T entries.

IV. Experimental Analysis

In this section we design and analyze experiments with the goal of answering the following questions:

Q1. Can MONET remove linear bias from topology embeddings and downstream prediction tasks?

Q2. Can MONET help reduce bias in topology embeddings even when they are used in nonlinear models?

Q3. Can MONET provide a reasonable trade-off between debiasing and accuracy on a downstream retrieval task?

Q4. Does adding the MONET unit to a method incur a significant performance overhead?

In all experiment settings, our topology embedding of interest for evaluation is the sum $W := U + V$ of the center-context topology embeddings U and V. All GloVe-based models are trained with TensorFlow [31] using the AdaGrad optimizer [32] with initial learning rate 0.05 and co-occurrence batch size 100. DeepWalk was trained using the gensim software [33]. Software for the MONET algorithm and code to reproduce the following experiments are publicly available[1].

A. *Experiment 1: Debiasing Blog Embeddings*

Here we seek to answer **Q1** and **Q2**, illustrating the effect of MONET debiasing by removing the effect of political affiliation from a blogger network [34]. The political blog network[2] has has 1,107 nodes corresponding to blog websites, 19,034 hyperlink edges between the blogs (after converting the graph to be undirected), and two clearly defined, equally sized communities of liberal and conservative bloggers.

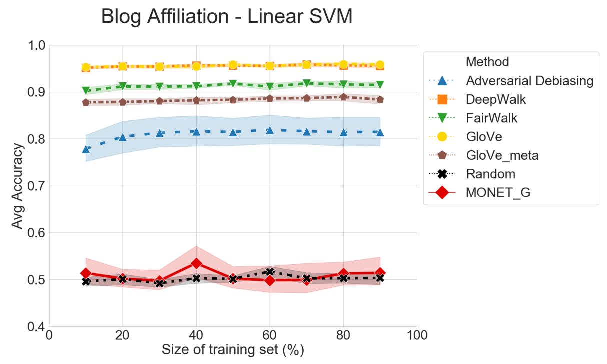
Baseline Methods. We compare MONET$_G$ against the following methods: (1) a random embedding generated from a multivariate Normal distribution; (2) DeepWalk; (3) GloVe; (4) GloVe$_{\text{meta}}$; (5) unsupervised adversarial debiasing modeled after [22], and (6) FairWalk [23]. We compare against GloVe and DeepWalk embeddings trained on FairWalk random walks. All methods use 16-dimensional topology embeddings. MONET and GloVe$_{\text{meta}}$ use 2-dimensional blog affiliation metadata embeddings. For all methods, 80 random walks of length 40 were generated for each node, and training batches were generated from size-10 random-walk context windows. All methods were trained for 5 epochs.

Evaluation Metrics. In this experiment we use two metrics to measure the bias of topology embeddings W:

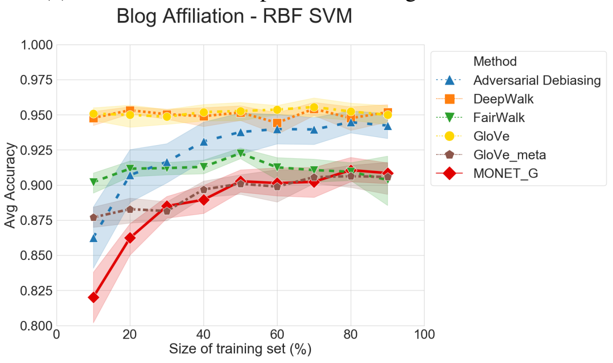
- Accuracy of a blog affiliation classifier trained on W.

[1]Software at https://github.com/google-research/google-research/tree/master/graph_embedding/monet

[2]Available within the Graph-Tool software [35]



(a) Sensitive attribute prediction using a linear model



(b) Sensitive attribute prediction using a non-linear model

Figure 4: Predictive accuracy of graph embeddings for political blogs. Lower is better (embeddings more debiased). MONET$_G$ achieves perfect debiasing with a linear classifier, due to its novel orthogonalization technique. Interestingly, on this task, MONET$_G$ is dominant even under a nonlinear classifier.

- Metadata Leakage $\mathcal{ML}(W, Z)$.

Because we are studying debiasing potential, Accuracy scores close to 0.5 are desirable (due to equally-sized blog groups), and Metadata Leakage equal to 0.0 is optimal. For methods without metadata embeddings, $\mathcal{ML}$ is computed with the metadata M.

Experiment Design. One repetition of the experiment proceeds as follows. In stage 1, random walks are generated, and all embedding methods are trained on the same set of walks (with the exception of FairWalk, which generates "fair" walks as described in Section 2.1). In stage 2, the evaluation metrics above are computed as follows. Metadata leakage is computed directly from embeddings. Classifier accuracy is computed over a range of training percentages using 5-fold cross validation (CV). Explicitly, for a given train set percentage $p \in \{10, 20, \ldots, 90\}$, we randomly sampled $p\%$ of the nodes for training. Then for each method, we train a predictive classifier with its embeddings for the training nodes, using 5-fold CV to optimize hyperparameters. Then we compute classifier accuracy on the held-out test set (identical across methods). We report mean and standard deviations across 10 independent repetitions of stage 1 + stage 2.

Classifier Accuracy. First we analyze the accuracy of a linear SVM at predicting political party affliation, shown in

Method	$\mathcal{ML}$	Metadata Importance Σ_T
GloVe	6559 ± 224	N/A
Adversary	4355 ± 383	N/A
DeepWalk	2684 ± 83	N/A
FairWalk	2335 ± 84	N/A
GloVe$_{\text{meta}}$	2065 ± 281	$\begin{pmatrix} .110 \pm .01 & -.106 \pm .01 \\ -.110 \pm .01 & .106 \pm .01 \end{pmatrix}$
Random	283 ± 22	N/A
MONET$_G$	$\mathbf{0.013} \pm 0.002$	$\begin{pmatrix} .187 \pm .01 & -.182 \pm .01 \\ -.187 \pm .01 & .182 \pm .01 \end{pmatrix}$

Table I: Only MONET removes political affiliation metadata leakage to precision error. Metadata Importance values - computed from layers of GloVe$_{\text{meta}}$ and MONET$_G$ - show that MONET allows stronger metadata learning.

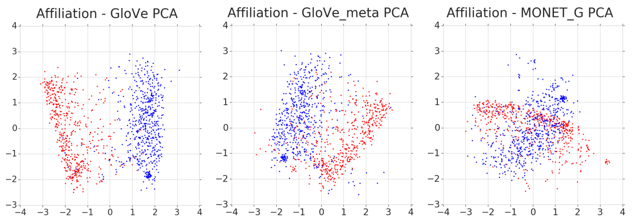


Figure 5: PCA of political blog graph embeddings. (a): affiliation separation clearly visible on standard GloVe embeddings. (b): affiliation separation reduces when GloVe$_{\text{meta}}$ captures some metadata information. (c): affiliation separation disappears with MONET$_G$ orthogonalized training.

Figure 4a. The performance of MONET$_G$ under the linear classifier is indistinguishable from the random baseline, which shows that the MONET unit perfectly debiased the GloVe embeddings during training, answering question **Q1** in the affirmative. This result is a direct implication of Theorem 2.

Interestingly, although the naive GloVe$_{\text{meta}}$ also has a metadata embedding partition, its performance under the linear classifier is much closer to the baselines with no metadata (GloVe and DeepWalk). This demonstrates the downstream effect of Theorem 1, and shows the necessity of desiderata D2 in a rigorous approach to graph embedding debiasing. We note that MONET also outperforms adversarial debiasing (our adaptation of [22] to the unsupervised GNN setting). Furthermore, using FairWalk random walks [23] eliminated almost no bias from the DeepWalk and GloVe baselines (only 2%-5% loss of predictive accuracy on the sensitive attribute). The reason for this is likely as follows: if there are some nodes with a neighbor set that represents only a small subset of sensitive attributes, then the FairWalk embedding of that node will remain biased toward those attributes. There are many nodes in the political blogs graph that only have neighbors of like political affiliation.

Second, we observe the results from training a non-linear classifier (the RBF SVM), as shown in Figure 4b. Here, we see that all embedding methods contain non-linear biases to exploit for the prediction of political affiliation. This includes MONET$_G$ because the MONET unit only performs linear debiasing. Surprisingly, here we find that MONET$_G$ still produces the *least* biased representations, even beating an adversarial approach. This answers question **Q2**, showing that MONET can reduce bias even when its representations are used as features inside of a non-linear model.

Leakage Results and Qualitative Analysis. Here we provide in-depth model understanding of MONET and, by comparison, GloVe$_{\text{meta}}$ and GloVe. The metadata leakage ($\mathcal{ML}$), as defined in Section III-B, is shown for each embedding set in Table I. We see that embedding models without metadata control have high leakage. As predicted by Corollary 1, GloVe$_{\text{meta}}$'s metadata leakage also remains large – whereas, due to Theorem 2, MONET$_G$'s $\mathcal{ML}$ is at machine precision. This again shows that a metadata embedding partition is not sufficient to isolate the metadata effect.

The need for MONET$_G$ over the naïve GloVe$_{\text{meta}}$ can be observed in two other ways. First, recall from Section III-D that the "Metadata Importance" matrix Σ_T encodes pairwise relationships between metadata dimensions in embedding space. There is a noticeable increase in Σ_T magnitude when MONET is used, implying that GloVe$_{\text{meta}}$ metadata embeddings are not capturing all possible metadata information. Second, Figure 5 shows the 2-D PCA of each models' 16-dimensional topology embeddings, colored by political affiliation. Clear separation by affiliation is visible with the GloVe model's PCA. The PCA of GloVe$_{\text{meta}}$ also shows separation, but less so. This shows that having a metadata partition in embedding space – desiderata D1 – does some work toward debiasing the topology dimensions. However, the orthogonalizing MONET unit does the bulk of the work, as shown by the overlap of the affiliation clusters in the MONET$_G$ PCA.

B. *Experiment 2: Debiasing a Shilling Attack*

In this experiment we investigate **Q3** and **Q4** in the context of using debiasing methods to defend against a *shilling attack* [36] on graph-embedding based recommender systems [37]. In a shilling attack, a number of users act together to artificially increase the likelihood that a particular influenced item will be recommended for a particular target item.

Data. In a single repetition of this experiment, we inject an artificial shilling attack into the MovieLens 100k dataset (files.grouplens.org). The raw data is represented as a bipartite graph with 943 users, 1682 movies ("items"), and a total of 100,000 ratings (edges). Each user has rated at least 20 items. At random, we sample 10 items into an influence set S_I, and a target item i_t to be attacked. We take a random sample of 5% of the existing users to be the set of attackers, $\mathcal{S}_A$. We then create a new graph, G_{attacked} which in addition to all the existing ratings, contains new ratings from each attacker $\in \mathcal{S}_A$ to each item $\in \mathcal{S}_I$ as well as the target video. As metadata, we use the per-item attacker rating count for each attacked item. However, to better simulate real-world scenarios, we only allow 50% (randomly sampled) attackers from the original 5% sample to be "known" when constructing these metadata.

Baseline Methods. We include all methods from the political blogs experiment in Section 4.1, with the exception of adversarial debiasing and FairWalk, as neither apply to continuous metadata. Instead, we applied a generalized correlation removal framework developed for removing word embedding bias [38, 26]. Specifically, we compute an "attack" direction (analogous to "gender" direction in language modeling literature) as follows. Let W be the GloVe topology embedding, and let M be the attacker metadata described above. Note that M_i is simply the scalar number of known attackers on item i. Then we compute the weighted attack direction as:

$$\vec{a} := \sum_{i:M_i>0} M_i \cdot W_i - \sum_{i:M_i=0} W_i \tag{8}$$

Following [38], we compute NLP debiased graph embeddings:

$$W_{\text{NLP}} := W - (W\vec{a}/||\vec{a}||)\vec{a}^T. \tag{9}$$

We also run relaxed MONET (see Section 3.3) for various values of λ, to investigate the bias and accuracy trade-off.

We generate 100 random walks (per node) of length 5 through the bipartite graph G_{attacked}. To embed only items and study item recommendation, we remove user nodes from the resultant random walks. The walk hyperparameters were chosen such that the (non-debiased) baseline GloVe model ran successfully enough to become biased by the shilling attack - and thus MONET$_G$ and other debiasing baselines would have attack signal to debias. All methods compute 128 dimensional topology embeddings for the items, trained over 20 epochs.

Evaluation Metrics. We compute two metrics to investigate bias-accuracy trade-off, averaged over 10 repetitions:

- **Bias**: Number of attacked items in S_I in top-20 embedding-nearest-neighbor list of i_t. This is a measure of *downstream* bias on a retrieval task.
- **Accuracy**: Mean reciprocal rank for item retrieval. Below, $NN(i)$ is the nearest neighbor of i by random-walk co-occurrence count and $rank_i(j)$ is the rank of item j against i's complete set of embedding cosine distances. In Figure 6, we show MRR/*random baseline* MRR, where $MRR = n^{-1}\sum_i rank_i(NN(i))^{-1}$.

Results. The results of the shilling experiment are shown in Figure 6, which compares the above metrics. First, we analyze number of attacked items, our measure of embedding bias. The topology embeddings from MONET$_G$ with $\lambda = 1.0$ prevent the most attacker bias by a large margin, letting less than 1 attacked item into the top-20 neighbors of t_i. This occurs even though the majority of observed co-occurrences for the algorithm had nothing to do with the attack in question, and only half of the true attackers were known. All other baselines left at least half of the attacked items in the top-20 list.

Next, we consider the ranking accuracy of each method. Here we observe that MONET$_G$ outperforms the random baseline by around 8.5x, and is comparable to one baseline (DeepWalk). Regarding the NLP debiasing method, we see that it outperformed MONET$_G$ in terms of ranking accuracy, but allowed a surprising number of attacked items (~7.5) in top-20 lists. We note that this method can not reduce bias further, and does offer any guarantees of performance. Finally, we observe that relaxing MONET's orthogonality constraint (λ, the level

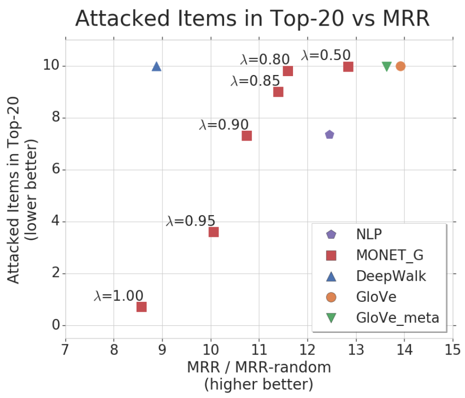


Figure 6: Shilling experiment accuracy (MRR) vs bias (Attacked Items). Points are averages over ten experiment runs. MONET$_G$ can completely prevent bias while maining 8x better accuracy than a random baseline.

Method	RSM r with W_{GloVe}	Wall Time (sec)
Random	0.031 ± 0.001	N/A
MONET$_G$	0.758 ± 0.004	442 ± 58
NLP	0.770 ± 0.008	357 ± 54
GloVe$_{\text{meta}}$	0.790 ± 0.002	359 ± 49
GloVe	1.000 ± 0.000	357 ± 54

Table II: Shilling experiment metrics. "RSM r with W_{GloVe}" is the Pearson correlation between the method's Representation Similarity Matrix (RSM) and GloVe RSM. NLP wall time is negligibly different from GloVe: it is a post-training correction.

of orthogonal projection), decreases the effective debiasing, but improves embedding performance (as might be expected). Note that $\lambda = 0.0$ (not shown), recovers the GloVe$_{\text{meta}}$ solution. This reveals a trade-off between embedding debiasing and prediction efficacy which has also been observed in other contexts [22]. This confirms **Q3** – that MONET allows for a controllable trade-off between debiasing and accuracy.

In addition to the ranking metrics, we also conduct Representational Similarity Analysis [39] of each embedding set against the standard GloVe embeddings. Specifically, we first compute the $n \times n$ row-wise representation cosine-similarity matrix (RSM) for each method. Then, we compute the Pearson correlation of the vectorized upper-diagonal entries of each RSM with those same entries from the GloVe RSM. This method has been used in recent deep learning literature to compare neural network layers [40]. We call this metric the "RSM r with W_{GloVe}", shown in Table II. Roughly, this measures embedding corruption incurred by debiasing procedures, using GloVe as a gold-standard. We find that while MONET$_G$ is most corrupted compared with GloVe$_{\text{meta}}$ and NLP, the difference is slight relative to the Random baseline, which is completely debiased but also fully corrupted and non-informative. This shows that the SVD operation does not harshly distort the baseline GloVe embedding space. We also report wall times for each method, showing that the metadata embedding and SVD operations of MONET$_G$ add negligibly to the runtime over GloVe. Together, these metrics answer **Q4** in the negative.

V. Conclusion

This work introduced MONET, a novel training technique which guarantees linear debiasing of graph embeddings from sensitive metadata. Our analyses illustrate the complexities of modeling metadata in GNNs, revealing that simply partitioning the embedding space to accommodate metadata does not achieve debiasing. In an empirical study, only MONET fully masked metadata signal from a linear classifier, and even outperformed adversarial techniques when used with a nonlinear model. A real-world item retrieval experiment showed that MONET successfully prevents shilling attack bias without severely impacting the performance of the underlying GNN.

There are two promising directions of future work in this new area. First, MONET only guarantees linear debiasing. Methods and exact guarantees for controlling nonlinear associations should be investigated. Toward this, the RBF SVM accuracy results on our political blogs experiment can serve as a benchmark. Second, we did not explore the performance of MONET in deeper or supervised GNNs, or its handling of high-dimensional metadata, each of which present interesting modeling challenges. For instance, in a deep supervised graph GNNs, it is not clear which hidden (embedding) layers would be worth debiasing, or whether associations between the metadata and prediction task are detrimental. Answering these questions will expand the technique's potential impact.

References

[1] I. Chami *et al.*, "Machine learning on graphs: A model and comprehensive taxonomy," *arXiv:2005.03675*, 2020.
[2] P. Cui *et al.*, "A survey on network embedding," *IEEE TKDE*, 2018.
[3] B. Perozzi *et al.*, "Deepwalk: Online learning of social representations," in *KDD*, 2014.
[4] J. Tang *et al.*, "LINE: Large-scale Information Network Embedding," in *WWW*, 2015.
[5] A. Grover and J. Leskovec, "node2vec: Scalable Feature Learning for Networks," in *KDD*, 2016.
[6] J. Qiu *et al.*, "Network Embedding as Matrix Factorization," in *WSDM*, 2018.
[7] Ş. Postăvaru *et al.*, "InstantEmbedding: Efficient local node representations," *arXiv:2010.06992*, 2020.
[8] L. Peel *et al.*, "Ground truth about metadata and community detection in networks," *Science Advances*, 2017.
[9] J. Halcrow *et al.*, "Grale: Designing networks for graph learning," in *KDD*, 2020.
[10] C. Yang *et al.*, "Network representation learning with rich text information," in *IJCAI*, 2015.
[11] M. E. Newman and A. Clauset, "Structure and inference in annotated networks," *Nature Communications*, 2016.
[12] Z. Yang *et al.*, "Revisiting semi-supervised learning with graph embeddings," in *ICML*, 2016.
[13] S. Abu-El-Haija *et al.*, "MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," *ICML*, 2019.
[14] H. Chen *et al.*, "A tutorial on network embeddings," *arXiv:1808.02590*, 2018.
[15] T. Mikolov *et al.*, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, 2013.
[16] J. Pennington *et al.*, "Glove: Global vectors for word representation," in *EMNLP*, 2014.
[17] R. Brochier, A. Guille, and J. Velcin, "Global vectors for node representations," in *WWW*, 2019.
[18] S. Zhu *et al.*, "Combining content and link for classification using matrix factorization." ACM, 2007.
[19] D. Zhang *et al.*, "Homophily, structure, and content augmented network representation learning," *ICDM*, 2016.
[20] C. Li *et al.*, "Ppne: property preserving network embedding," in *DASFAA*, 2017.
[21] J. Guo *et al.*, "Enhancing network embedding with auxiliary information," in *DASFAA*, 2018.
[22] A. J. Bose and W. L. Hamilton, "Compositional fairness constraints for graph embeddings," *ICML*, 2019.
[23] T. Rahman *et al.*, "Fairwalk: towards fair graph embedding," in *IJCAI*, 2019.
[24] J. Zhao *et al.*, "Learning gender-neutral word embeddings," *EMNLP*, 2018.
[25] A. Caliskan *et al.*, "Semantics derived automatically from language corpora contain human-like biases," *Science*, 2017.
[26] T. Bolukbasi *et al.*, "Man is to computer programmer as woman is to homemaker?" in *NeurIPS*, 2016.
[27] Y. He *et al.*, "A geometric solution to fair representations," in *AIES*, 2020.
[28] M. Hardt *et al.*, "Equality of opportunity in supervised learning," in *NeurIPS*, 2016.
[29] B. H. Zhang *et al.*, "Mitigating unwanted biases with adversarial learning," in *AAAI*. ACM, 2018.
[30] L. N. Trefethen and D. Bau III, *Numerical linear algebra*, 1997.
[31] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *USENIX OSDI*, 2016.
[32] J. Duchi *et al.*, "Adaptive subgradient methods for online learning and stochastic optimization," *JMLR*, 2011.
[33] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *LREC*. ELRA, 2010.
[34] L. Adamic and N. Glance, "The political blogosphere and the 2004 US election: divided they blog," *IWLD*, 2005.
[35] T. P. Peixoto, "The graph-tool python library," http://figshare.com/articles/graph_tool/1164194, 2014.
[36] P.-A. Chirita *et al.*, "Preventing shilling attacks in online recommender systems," *WIDM*, 2005.
[37] R. Ying *et al.*, "Graph convolutional neural networks for web-scale recommender systems," in *KDD*. ACM, 2018.
[38] B. Schmidt, "Rejecting the gender binary: a vector-space operation," *Ben's Bookworm Blog*, 2015.
[39] N. Kriegeskorte *et al.*, "Representational similarity analysis-connecting the branches of systems neuroscience," *Frontiers in systems neuroscience*, 2008.
[40] A. Morcos *et al.*, "Insights on representational similarity in neural networks with canonical correlation," in *NeurIPS*, 2018.