MONET: DEBIASING GRAPH EMBEDDINGS VIA THE METADATA-ORTHOGONAL TRAINING UNIT

Anonymous authors
Paper under double-blind review

ABSTRACT

Are Graph Neural Networks (GNNs) fair? In many real world graphs, the formation of edges is related to certain node attributes (e.g. gender, community, reputation). In this case, standard GNNs using these edges will be biased by this information, as it is encoded in the structure of the adjacency matrix itself. In this paper, we show that when metadata is correlated with the formation of node neighborhoods, unsupervised node embedding dimensions learn this metadata. This bias implies an inability to control for important covariates in real-world applications, such as recommendation systems.

To solve these issues, we introduce the Metadata-Orthogonal Node Embedding Training (MONET) unit, a general model for debiasing embeddings of nodes in a graph. MONET achieves this by ensuring that the node embeddings are trained on a hyperplane orthogonal to that of the node metadata. This effectively organizes unstructured embedding dimensions into an interpretable topology-only, metadata-only division with no linear interactions. We illustrate the effectiveness of MONET though our experiments on a variety of real world graphs, which shows that our method can learn and remove the effect of arbitrary covariates in tasks such as preventing the leakage of political party affiliation in a blog network, and thwarting the gaming of embedding-based recommendation systems.

1 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 (10). While graph embeddings can be estimated directly by unsupervised algorithms using the graph's structure (e.g. 22; 26; 13; 23), there is often additional (non-relational) information available for each node in the graph. This information, frequently referred to as node *attributes* or node *metadata*, can contain information that is useful for prediction tasks including demographic, geo-spatial, and/or textual features.

The interplay between a node's metadata and edges is a rich and active area of research. Interestingly, in a number of cases, this metadata can be measurably related to a graph's structure (19), and in some instances there may be a causal relationship (the node's attributes influence the formation of edges). As such, metadata can enhance graph learning models (28; 18), and conversely, graphs can be used as regularizers in supervised and semi-supervised models of node features (29; 11). Furthermore, metadata are commonly used as evaluation data for graph embeddings (8). For example, node embeddings trained on a Flickr user graph were shown to predict user-specified Flickr "interests" (22). This is presumably because users (as nodes) in the Flickr graph tend to follow users with similar interests, which illustrates a potential causal connection between node topology and node metadata.

However, despite the usefulness and prevalence of metadata in graph learning, there are instances where it desirable to design a system to *avoid* the effects of a particular kind of *sensitive* data. For instance, the designers of a recommendation system may want to make recommendations independent of a user's demographic information or location.

At first glance, this may seem like an artificial dilemma – surely one could just avoid the problem by not adding such sensitive attributes to the model. However, such an approach (ignoring a sensitive attribute) does not control for any existing correlations that may exist between the sensitive metadata

and the edges of a node. 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. Surprisingly, almost all of the existing work in the area (28; 31) has ignored this important realization.[1]

In this work, we seek to refocus the discussion about graph learning with node metadata. To this end, we propose a novel, general technique for extending graph representations with metadata embedding dimensions while debiasing the remaining (topology) dimensions. Specifically, our contributions are the following:

1. The Metadata-Orthogonal Node Embedding Training (MONET) unit, a novel GNN algorithm which jointly embeds graph topology and graph metadata while enforcing independence between the two embedding spaces.
2. Analysis which proves that a naive approach (adding metadata embeddings without MONET) leaks metadata information into topology embeddings, and that the MONET unit does not.
3. Experimental results on real world graphs which show that MONET can successfully "debias" topology embeddings while relegating metadata information to separate metadata embeddings.

2 PRELIMINARIES

Early graph embedding methods involved dimensionality reduction techniques like multidimensional scaling and singular value decomposition (8). In this paper we use graph neural networks trained on random walks, similarly to DeepWalk (22). 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 (17) or GloVe (21)) to learn low dimensional representations that preserve the observed co-occurrence similarity.

Let W be a d-dimensional graph *embedding* matrix, $W \in \mathbb{R}^{n \times d}$, 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. As a concrete example, in this paper we consider the debiasing of a recently proposed graph embedding using the GloVe model (6). Its training objective is:

$$\text{GloVe}(U, V, a, b|C) = \sum_{i,j \leq n} f_\alpha(C_{ij})(a_i + b_j + U_i^T V_j - \log(C_{ij}))^2, \tag{1}$$

where $U, V \in \mathbb{R}^{n \times d}$ are the "center" and "context" embeddings, $a, b \in \mathbb{R}^{n \times 1}$ are the biases, C is the walk-distance-weighted context co-occurrences, and f_α is the loss smoothing function (21). We use the GloVe model throughout to demonstrate topology/metadata embeddings and metadata-orthogonal training, though the MONET unit we propose is broadly generalizable.

Notation. In this paper, given a matrix $A \in \mathbb{R}^{n \times d}$ 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.

3 METADATA EMBEDDINGS AND ORTHOGONAL TRAINING

In this section we present MONET, our proposed method for separating and controlling the effects of metadata on topology embeddings. First, we begin by outlining the straightforward extension of metadata to traditional embedding models in Section 3.1. Next, in Section 3.2, we prove that such a simple model will leak information from the metadata to the topology (structural) embeddings. Then, in Section 3.3 we present MONET, our proposed approach for training embeddings of a graph's structure which are not correlated with metadata. Finally, we conclude with some analysis of MONET in Section 3.4

[1] While preparing this manuscript, we have become aware of a recent independent result (5) in this area. In contrast to that work, we use a substantially different methodology which offers guarantees about the debiasing process.

3.1 JOINTLY MODELING METADATA & TOPOLOGY

A natural first approach to modeling the effects of metadata on the graph is to explicitly include the node metadata as part of a node embedding model. For instance, to extend Eq. (1), in addition to U and V (the "topology embeddings"), we can consider the node metadata M directly ($M \in \mathbb{R}^{n \times m}$, row vector M_i is the metadata for node u_i). We then can define metadata embeddings $X = MT_1$, $Y = MT_2$, where T_1, T_2 are trainable transformations, and propose the concatenations $[U, X]$ and $[V, Y]$ as full-graph representations. The GloVe loss with metadata embeddings is:

$$\text{GloVe}_{meta}(U, V, T_1, T_2, a, b|C, M) = \frac{1}{2} \sum_{i,j \leq n} f_\alpha(C_{ij})(a_i + b_j + U_i^T V_j + X_i^T Y_j - \log(C_{ij}))^2. \quad (2)$$

While in this paper we demonstrate metadata embeddings within the GloVe model, they can be incorporated in any dot-product-based graph neural network. For instance, the well-known DeepWalk (22) loss, which is based on word2vec (17), would incorporate metadata embeddings as follows:

$$\text{DeepWalk}_{meta}(U, V, T_1, T_2|\mathcal{W}, M) = -\sum_{i,j \in \mathcal{W}} \log(U_i^T V_j + X_i^T Y_j) - \sum_{k \in K_i} \log(-U_i^T V_k - X_i^T Y_k). \quad (3)$$

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. For GloVe, DeepWalk, and many other GNNs, this approach augments the overall graph representation by concatenating metadata-learned dimensions.

However, this naïve approach does not guarantee that the topology embeddings converge to be statistically independent of the metadata embeddings. Suppose that the metadata (like demographic information) are indeed associated with the formation of links in the graph. In this case, any algorithm which does not explicitly model and remove the association will be biased as a result of it. In the next section we formalize this concept, which we call *metadata leakage*.

3.2 METADATA LEAKAGE IN GRAPH NEURAL NETWORKS

Here, we formally define *metadata leakage* for general topology and metadata embeddings, and show how it can occur even in embedding models with separate metadata embeddings. All proofs appear in the Appendix.

Definition 1. *The **metadata leakage** of metadata embeddings $Z \in \mathbb{R}^{n \times d_Z}$ into topology embeddings $W \in \mathbb{R}^{n \times d}$ is defined $\mathcal{ML}(Z, W) := ||Z^T W||_F^2$. 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 that explicitly include the metadata, like Eqs. (2) and (3). To demonstrate this, we consider for simplicity a reduced metadata-aware GloVe loss with $W := U = V \in \mathbb{R}^{n \times d}$ as the sole topology embedding and $T := T_1 = T_2 \in \mathbb{R}^{m \times d_Z}$ as the sole metadata transformation parameter. With $Z := MT$, the reduced loss is:

$$\text{GloVe}^*_{meta}(W, T, a|C, M) = \frac{1}{2} \sum_{i,j \leq n} (a_i + a_j + W_i^T W_j + Z_i^T Z_j - \log(C_{ij}))^2 \quad (4)$$

We now show that under a random update of the GloVe^*_{meta} model in Eq. (4), the expected metadata leakage is non-zero. Specifically, let (i, j) be a node pair from 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 $B \in \mathbb{R}^{m \times d_B}$, and define ground-truth metadata embeddings $\tilde{Z} := MB$, which represent the "true" dimensions of the metadata effect on the co-occurrences 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 $\mu_W := \mathbb{E}[W_i]$ and $\Sigma_W := \mathbb{E}[W_i W_i^T]$. Define μ_M, Σ_M similarly. Then our main Theorem is as follows:

Theorem 1. *Assume $\Sigma_W = \sigma_W I_d$ for $\sigma_W > 0$, $\mu_W = \mathbf{0}_{d \times 1}$, and $\mu_M = \mathbf{0}_{m \times 1}$. Suppose for some fixed $\theta \in \mathbb{R}$ 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 $\mathbb{E}[M_i W_i^T] = \beta \in \mathbb{R}^{m \times d}$, we have*

$$\mathbb{E}[\mathcal{ML}(Z, W')] \geq 2||T^T \left[\Sigma_M(\Sigma_B - \Sigma_T) + (n - \sigma_W) I_m\right] \beta||_F^2. \quad (5)$$

Algorithm 1 MONET Unit Training Step

Given: 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 $[W^\perp, Z]$
procedure BACKWARD PASS DEBIASING(δ_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 $W^\perp$

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

Corollary 1. *Under the assumptions of Theorem 1,* $\mathbb{E}[\mathcal{ML}(Z, W)] = \Omega(n||T^T\beta||_F^2)$ *as* $n \to \infty$.

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

3.3 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 is a two-step algorithm applied to the training of a topology embedding in a neural network, and is detailed in Algorithm 1. The input to a MONET unit is a metadata embedding $Z \in \mathbb{R}^{n\times d_z}$ and a target topology embedding $W \in \mathbb{R}^{n\times d}$ for debiasing. Then, let Q_Z be the left-singular vectors of Z, and define the projection $P_Z := I_{n\times n} - Q_Z Q_Z^T$. In the forward pass procedure, 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 $\delta_W^\perp = P_Z \delta_W$. Figure 1 illustrates a geometric interpretation of the MONET algorithm.

Straightforward properties of the SVD show that MONET directly prevents metadata leakage:

Theorem 2. *Using Algorithm 1,* $\mathcal{ML}(Z, W^\perp) = 0$ *and* $\mathcal{ML}(Z, \delta^\perp) = 0$.

We note that in this work we have only considered *linear* metadata leakage; 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). We denote this model MONET$_G$, and create it as follows. 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 (21). Figure 2 shows an illustration of MONET$_G$.

3.4 ANALYSIS

Here we address some brief remarks 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 $O(nd_z^2)$ (27). 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 $O(nd_z \max\{d, d_z\})$.

Metadata Parameter Interpretation. The i, j terms in the sum of the loss for GloVe models with metadata (GloVe$_{meta}$ and MONET$_G$) involve the dot product $X_i^T Y_j = M_i^T T_1 T_2^T M_j$. That expan-

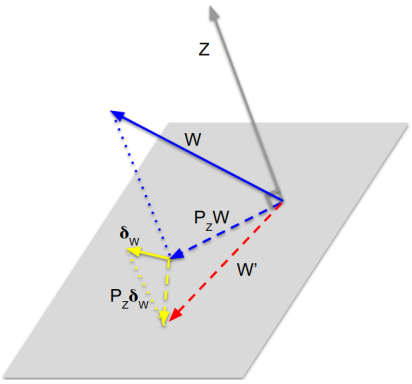


Figure 1: Geometric interpretation of MONET. Both prediction and training for W occur on a hyperplane orthogonal to Z. In the forward pass, W is projected onto the Z-orthogonal plane. When an update δ_W is proposed, it too is projected, resulting in the best metadata-orthogonal update. This allows W to explore the space of unknown latent structure without bias from Z.

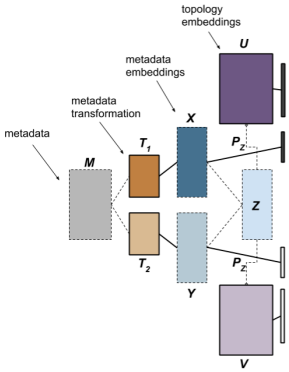


Figure 2: Illustration of MONET$_G$. U and V are topology embeddings. The MONET unit adds a feed-forward transformation of the metadata, resulting in metadata embeddings X and Y. $Z = X + Y$ gives the combined metadata representation, used to debias U and V via P_Z. Dotted lines indicate stopped gradient flow during backpropagation.

sion 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 entires.

4 Metadata De-Biasing Experiments

Here we empirically demonstrate Theorems 1 and 2 by confirming the following hypotheses:

1. H1. The MONET unit can remove leakage of metadata information from topology embeddings, so that the topology embeddings cannot predict the metadata.
2. H2. The MONET unit can make recommender systems more robust to abuse by removing malicious user directions from rating graphs.

For all embedding models, we use the center-context embedding sum of topology embeddings $W := U + V$ as the graph representation for task evaluation. Note that some standard baselines (e.g. DeepWalk) do not incorporate metadata and therefore only train topology embeddings. All GloVe-based models are trained with TensorFlow (1) using the AdaGrad optimizer (12) with initial learning rate 0.05. DeepWalk models were trained using the gensim software (24).

4.1 Quantitative Experiment: Political Blogs Network

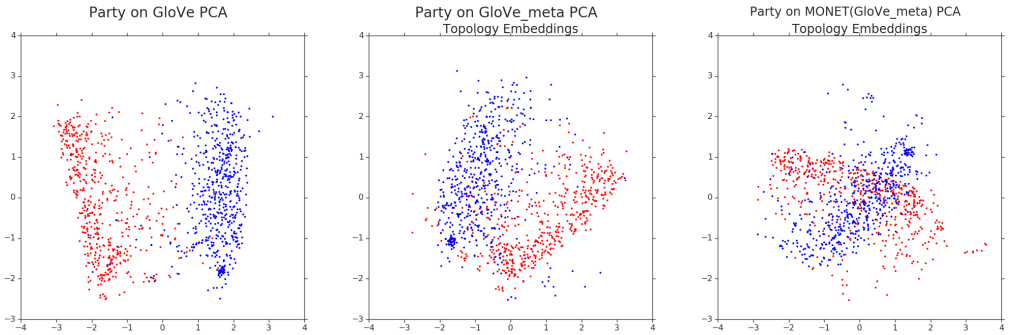
To address H1, illustrating Theorem 1 and the effect of MONET de-biasing, we embed the effect of political ideology on a blogger network (3). The political blog network[2] has has 1,107 nodes corresponding to blog websites, 19,034 hyperlink edges between the blogs (after converting the graph to be undirected), and two clearly defined, equally sized communities of liberal and conservative bloggers.

Design and Methods. In this experiment all graph neural network models were trained on 5 iterations across 80 random walks per node of length 40 with context window size 10. Topology embeddings had dimension 16, and metadata embeddings had dimension 2. We measure embedding bias by the Macro-F1 score of a linear SVM predicting political party from the embeddings, using a LIBLINEAR

[2] Available within the Graph-Tool software (20)

Model	F1 (mean $\pm$ std)	$\Sigma_T = T_1 T_2^T$ (mean $\pm$ std)	$\mathcal{ML}$
DeepWalk	95.59% $\pm$0.07%	N/A	2743.9 ± 36.7
GloVe	95.94% $\pm$0.07%	N/A	6598.0 ± 200.1
GloVe$_{meta}$	88.33% $\pm$0.60%	$\begin{pmatrix} 0.108 \pm 0.006 & -0.106 \pm 0.004 \\ -0.108 \pm 0.009 & 0.106 \pm 0.006 \end{pmatrix}$	1827.6 ± 289.7
MONET$_G$	**49.30%** $\pm$0.60%	$\begin{pmatrix} 0.180 \pm 0.006 & -0.178 \pm 0.006 \\ -0.181 \pm 0.008 & 0.179 \pm 0.006 \end{pmatrix}$	0.018 ± 0.002

Table 1: Macro-F1 scores from political blog network classifications using graph topology embeddings only. MONET is successful in removing all metadata information from the topology embeddings – the links in the graph are no longer an effective predictor of political party. Comparison of the metadata transformation product Σ_T between GloVe$_{meta}$ and MONET$_G$ shows MONET allows for considerably more metadata information learning. Finally, only MONET removes metadata leakage to precision error (recall $\mathcal{ML}()$ is a Frobenius norm).



(a) (b) (c)

Figure 3: PCA of political blog graph embeddings. (a): Party separation clearly visible on standard GloVe embeddings. (b): Party separation reduces when GloVe$_{meta}$ captures some metadata information. (c): Party separation disappears with MONET$_G$ orthogonalized training.

implementation (7). For each embedding set, we compute the mean and standard deviation Macro-F1 over 10 independent classification repetitions, each trained using half of the node labels sampled at random. To assess metadata information leakage, we also track the metadata dimension importance matrix $\Sigma_T := T_1 T_2^T$, recalling its interpretation from Section 3.

Results. Table 1 shows that the baselines DeepWalk and GloVe are highly effective at predicting political party, and therefore biased. This is unsurprising, as these methods are trained without metadata information, and were originally intended to encode low-dimensional structure like that present in this data set. The bias in DeepWalk and GloVe embeddings is further seen in their metadata leakage values, computed using political party one-hot vectors as metadata embeddings.

Considering the embedding models with metadata embeddings, we find that, interestingly, GloVe$_{meta}$'s topology embeddings are still able to predict political party with 88.3% Macro-F1. Also, as predicted by Corollary 1, GloVe$_{meta}$'s metadata leakage remains $O(n)$. This shows that simply concatenating metadata embeddings is not sufficient to isolate the metadata effect. In contrast, MONET$_G$ achieves random Macro-F1 and no metadata leakage (under machine precision), demonstrating that on this data, the MONET unit is necessary to de-bias the blog embeddings from political party. This contrast is seen in two other ways. First, there is a noticeable increase in Σ_T magnitude when MONET is used, implying that GloVe$_{meta}$ metadata embeddings are not capturing all possible metadata information. Second, as seen in Fig. 3, the 2-dimensional PCA plots of the GloVe$_{meta}$ embeddings still show political party separation, whereas the MONET$_G$ PCA dimensions reveal strong mixing.

Model	Manipulated Items in Top-20 (mean $\pm$ std dev)	Embedding Distance Correlation w/GloVe
DeepWalk	9.9 ± 0.40	0.385 ± 0.005
GloVe	9.8 ± 0.76	1.000 ± 0.000
GloVe$_{meta}$	9.8 ± 0.476	0.852 ± 0.002
NLP Debiasing (25; 4) (sum)	8.1 ± 1.7	0.566 ± 0.004
NLP Debiasing (25; 4) (max)	4.9 ± 3.4	$\mathbf{0.990} \pm 0.003$
MONET$_G$	$\mathbf{1.2} \pm 1.7$	0.831 ± 0.004

Table 2: Results from the shilling attack experiment. Attackers attempt to insert 10 items in the top-20 recommendations of a target video. The results show that MONET can best mitigate the effect of an attack under incomplete information. We note that there is an implicit trade-off between de-biasing and maintaining correlation with the original (biased) embeddings.

4.2 Experiment 2: Thwarting Attacks on Graph-based Recommendation Systems

In this experiment we address H2, investigating the effectiveness of MONET to defend against a *shilling attack* (9) against graph-embedding based recommender systems (30). 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[3]. The raw data is represented as a bipartite graph with 943 users, 1682 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. (Note that this corresponds to several varieties of behavior including both incentivizing formerly good users, and account takeover.)

Design and Methods. For each embedding method, we perform random walks through the new bipartite graph G_{attacked}. As we wish to study item recommendation, in the random walks, we simply remove user nodes each time they are visited (so the walks contain only pairwise co-occurrence information over items). With any given network embedding, we measure its bias by the number of influence items in S_I in top-20 embedding-nearest-neighbor list of i_t. As metadata, we allow MONET$_G$ to know the per-movie attacker rating count for each attacked movie. However, to better demonstrate real-world performance, we only allow 50% (randomly sampled) attackers from the original 5% sample to be "known" when constructing these metadata. As non-debiasing baselines, we compare against DeepWalk and Glove. As de-biasing baselines, we applied a generalized correlation removal framework developed for removing word embedding bias (25; 4). Specifically, we tried two approaches to "debias" the GloVe embedding of the MovieLens graph – as the "gender" embedding direction, we tried both (a) the most attacked movie vector and (b) the sum of attacked movie vectors. All methods use 128 dimensional topology embeddings and are trained on 100 random walks per node, each walk of length 5.

Results. As seen in Table 2, the topology embeddings from MONET$_G$ are the least biased by a large margin, letting on average only 1.2 influence items in the top-20 neighbors of t_i. Interestingly, we note that this behavior occurs even though the majority of observed co-occurrences for the algorithm had nothing to do with the attack in question, and only the known 50% of attackers were used to construct the metadata. In contrast, all baselines (including those that explicitly model the attacker metadata) left at least around half of the attacked items in the top-20 list. To measure the extent to which de-biased embeddings retain the original recommendation signal, we compute the pair-wise embedding distances of each method, and compute their Pearson correlation to the standard GloVe embeddings. We find that MONET embedding distances achieve high correlation (0.83) to the original distances, showing that with MONET it is possible to nearly nullify a shilling attack while preserving most of the signal from the true, un-attacked ratings. We note that the max-attack-embedding baseline higher embedding distance correlation, but this method let many more attacked items (on average) into higher ranks. This reveals a trade-off between embedding de-biasing and prediction efficacy which has also been observed in other contexts (30).

[3]Available: `http://files.grouplens.org/datasets/movielens/ml-100k/`

5 RELATED WORK

Though graph learning is an immense field, a minority of unsupervised graph embedding techniques involve graph metadata. To our knowledge, none of these techniques involve either metadata orthogonalization or the capacity to learn arbitrary metadata transformations. (32) is a matrix factorization approach which uses a shared node embedding matrix to factor both the graph adjacencies and the raw metadata in a joint loss, with a tunable parameter to control the influence of the metadata loss. Similarly, (16) pre-computes a metadata similarity matrix and trains shared center-context embedding matrices on the metadata similarities and random walk similarities. In contrast, we learn the direction and effect of metadata as neural network parameters, and we separate those parameters into unique embedding dimensions. (28) and (31) are matrix factorization approaches which factor an approximation to the co-occurrence matrix into *equally-sized* metadata and topology embeddings, and were built mainly for text metadata. Their approaches enforce metric space similarity and dimensional homogeneity between metadata and topology representations, restrictions that we do not rely on and are ill-suited to the setting with multiple types of arbitrarily-sized metadata. (14) constructs random walks that traverse between the original graph and the metadata freely, an approach which runs counter to our ability to separate out the effects of metadata on graph adjacencies. (18) introduce a version of the stochastic block model with metadata priors, and show that the estimated posteriors yield insight into the influence of metadata on the graph. However, this model estimates a community partition and in/out-community probabilities - it does not yield embeddings either of the node topology or the node metadata. There has been work in Natural Language Processing on removing gender bias from word embeddings (e.g. 4), but these methods operate with pre-computed embeddings and rely on identification of gendered terminology.

Additionally, there has been a wealth of work studying semi-supervised learning with graphs (e.g. 29) and graph convolutional networks (e.g. 2; 15; 11), which use graph metadata as features. While most semi-supervised and supervised neural networks for graphs indirectly produce embeddings that in some cases can be identified with feature and topology dimensions, they are trained as part of prediction or label propagation tasks. Therefore, the topology embeddings are free to correlate with features to the extent that this serves the loss function - there is no explicit separation of topology and metadata dimensions. In this paper, we have studied the benefits of metadata orthogonalization in the unsupervised setting, and we leave the exploration of our techniques in the semi-supervised and supervised settings to future work.

6 CONCLUSION

In this work, we have shown that unsupervised training of graph embeddings induces bias from important graph metadata. We proposed a novel solution to address this problem – the Metadata-Orthogonal Node Embedding Training (MONET) unit. The MONET unit is the first graph learning technique for training-time debiasing of embeddings, using orthogonalization. Our experimental results using real datasets showed that MONET is able to encode the effect of graph metadata in isolated embedding dimensions, and simultaneously remove the effect from other dimensions. This has immediate practical applications, which we illustrate by mitigating a simulated shilling attack on a real dataset of movie ratings.

This work was meant to introduce the basic principles underlying the need for the MONET technique, and show its utility in a shallow graph neural network (GloVe). While we used a shallow network for instructional purposes, we note that MONET is generalizable, and MONET units can be used to debias any set of embeddings from another set during training. Subsequent research can explore the use of MONET in deeper networks and potentially semi-supervised models or graph convolutional networks. As MONET's SVD calculation can be expensive with large graphs and large embedding dimensions, future research could be in assessing the effect of SVD approximations, or training algorithms that utilize caching of previous metadata embedding SVDs to speed up training.

REFERENCES

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX}*

Symposium on Operating Systems Design and Implementation ({OSDI} 16), pages 265–283, 2016.

[2] S. Abu-El-Haija, B. Perozzi, A. Kapoor, H. Harutyunyan, N. Alipourfard, K. Lerman, G. V. Steeg, and A. Galstyan. Mixhop: Higher-order graph convolution architectures via sparsified neighborhood mixing. *arXiv preprint arXiv:1905.00067*, 2019.

[3] L. A. Adamic and N. Glance. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, pages 36–43. ACM, 2005.

[4] T. Bolukbasi, K.-W. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*, pages 4349–4357, 2016.

[5] A. J. Bose and W. L. Hamilton. Compositional fairness constraints for graph embeddings, 2019.

[6] R. Brochier, A. Guille, and J. Velcin. Global vectors for node representations. *arXiv preprint arXiv:1902.11004*, 2019.

[7] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

[8] H. Chen, B. Perozzi, R. Al-Rfou, and S. Skiena. A tutorial on network embeddings. *arXiv preprint arXiv:1808.02590*, 2018.

[9] P.-A. Chirita, W. Nejdl, and C. Zamfir. Preventing shilling attacks in online recommender systems. In *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management*, WIDM '05, pages 67–74, New York, NY, USA, 2005. ACM. ISBN 1-59593-194-5. doi: 10.1145/1097047.1097061. URL http://doi.acm.org/10.1145/1097047.1097061.

[10] P. Cui, X. Wang, J. Pei, and W. Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[11] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[12] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

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

[14] J. Guo, L. Xu, X. Huang, and E. Chen. Enhancing network embedding with auxiliary information: An explicit matrix factorization perspective. In *International Conference on Database Systems for Advanced Applications*, pages 3–19. Springer, 2018.

[15] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[16] C. Li, S. Wang, D. Yang, Z. Li, Y. Yang, X. Zhang, and J. Zhou. Ppne: property preserving network embedding. In *International Conference on Database Systems for Advanced Applications*, pages 163–179. Springer, 2017.

[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[18] M. E. Newman and A. Clauset. Structure and inference in annotated networks. *Nature communications*, 7:11863, 2016.

[19] L. Peel, D. B. Larremore, and A. Clauset. The ground truth about metadata and community detection in networks. *Science advances*, 3(5):e1602548, 2017.

[20] T. P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare.1164194. URL http://figshare.com/articles/graph_tool/1164194.

[21] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[22] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[23] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 459–467. ACM, 2018.

[24] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[25] B. Schmidt. Rejecting the gender binary: a vector-space operation. *Ben's Bookworm Blog*, 2015.

[26] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[27] L. N. Trefethen and D. Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[28] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang. Network representation learning with rich text information. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[29] Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 40–48. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045396.

[30] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983. ACM, 2018.

[31] D. Zhang, J. Yin, X. Zhu, and C. Zhang. Homophily, structure, and content augmented network representation learning. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 609–618. IEEE, 2016.

[32] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 487–494. ACM, 2007.

A APPENDIX

Proposition 1. *Under the assumptions of Theorem 1, we have*

$$\mathbb{E}[Z^T \delta_W(i,j)] = 2\left[\Sigma_M(\Sigma_B - \Sigma_T) + \sigma_W I_m\right]\beta. \tag{6}$$

Proof. Derivatives of GloVe$^*_{meta}$ yield that the i-th row of $\delta_W(i,j)$ is $d_{ij}W_j^T$, where

$$\begin{aligned} d_{ij} &= \log(C_{ij}) - Z_i^T Z_j - W_i^T W_j - a_i - a_j \\ &= \theta + \tilde{Z}_i^T \tilde{Z}_j - Z_i^T Z_j - W_i^T W_j - a_i - a_j. \end{aligned} \tag{7}$$

Similarly the j-th row is $d_{ij}W_i^T$, and all other rows are zero vectors. Hence

$$\mathbb{E}[Z^T\delta_W(i,j)] = \mathbb{E}[Z_i d_{ij} W_j^T] + \mathbb{E}[Z_j d_{ij} W_i^T]. \tag{8}$$

We derive the second term on the right-hand side of Equation 8; the first term follows by symmetry. Note first that $\mathbb{E}Z_i(\theta - a_i - b_j)W_j^T = 0$ by independence and centering assumptions. Second:

$$\mathbb{E}[Z_i W_i^T W_j W_j^T] = T^T\mathbb{E}[M_i W_i^T W_j W_j^T] = T^T\mathbb{E}[M_i W_i^T]\mathbb{E}[W_j W_j^T] = T^T\beta\sigma_W I_d = T^T\sigma_W I_m\beta$$

by independence. Third:

$$\mathbb{E}[Z_i Z_i^T Z_j W_j^T] = T^T\mathbb{E}[M_i M_i^T T T^T M_j W_j^T] = T^T\left(\mathbb{E}[M_i M_i^T]\right)TT^T\left(\mathbb{E}[M_j W_j^T]\right) = T^T\Sigma_M\Sigma_T\beta$$

by independence, and similarly $\mathbb{E}[Z_i\tilde{Z}_i^T\tilde{Z}_j W_j^T] = T^T\Sigma_M\Sigma_B\beta$. Combining these with Equation 7, we have

$$\mathbb{E}[Z_i d_{ij} W_j^T] = T^T\left[\Sigma_M(\Sigma_B - \Sigma_T) - \sigma_W I_d\right]\beta. \tag{9}$$

Applying symmetry to the second term in Equation 8 completes the proof. □

A.1 Proof of Theorem 1

Proof. Proposition 1 gives $\mathbb{E}[Z^T\delta_W(i,j)] = 2T^T\left[\Sigma_M(\Sigma_B - \Sigma_T) + \sigma_W I_m\right]\beta$. Second, note that $\mathbb{E}[M_i W_i^T] = \beta \Rightarrow M^T W = n\beta$ and thus $Z^W = T^T M^T W = nT^T\beta$. Recalling that $W' = W + \delta_W(i,j)$ we have

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

Applying Jensen's Inequality completes the proof. □

A.2 Proof of Theorem 2

Proof. Consider metadata embeddings $Z \in \mathbb{R}^{n\times d_Z}$ and, as in the MONET algorithm, define the projection $P_Z = I_{d_Z} - Q_Z Q_Z^T$, where Q_Z are the left-singular vectors of Z. By properties of the SVD, $Z^T Q_Z Q_Z^T = Z_T$, and hence $Z^T P_Z = \mathbf{0}_{n\times d_Z}$. This means that $Z^T W^\perp = Z^T\delta_W^\perp = \mathbf{0}_{d_Z\times d}$, which completes the proof by definition of metadata leakage. □