FOBE and HOBE: First- and High-Order Bipartite Embeddings

Justin Sybrandt
School of Computing
Clemson University
Clemson, SC 29631
jsybran@clemson.edu

Ilya Safro
School of Computing
Clemson University
Clemson, SC 29631
isafro@clemson.edu

Abstract

Typical graph embeddings may not capture type-specific bipartite graph features that arise in such areas as recommender systems, data visualization, and drug discovery. Machine learning methods utilized in these applications would be better served with specialized embedding techniques. We propose two embeddings for bipartite graphs that decompose edges into sets of indirect relationships between node neighborhoods. When sampling higher-order relationships, we reinforce similarities through algebraic distance on graphs. We also introduce ensemble embeddings to combine both into a "best of both worlds" embedding. The proposed methods are evaluated on link prediction and recommendation tasks and compared with other state-of-the-art embeddings. Our embeddings are found to perform better on recommendation tasks and equally competitive in link prediction. While being all highly beneficial in applications, we demonstrate that none of the existing state-of-the-art or our embeddings is clearly superior (in contrast to what is claimed in many papers), and discuss the trade offs present among them.
Reproducibility: Our code, data sets, and results are all publicly available online at: `http://bit.ly/fobe_hobe_code`[1].

1 Introduction

Graph embedding methods place nodes into a continuous vector space in order to capture structural properties that enable machine learning tasks [11]. While many have made significant progress embedding general graphs [22, 27, 12], we find that while bipartite graphs have been the focus of some study [10], the field is far from settled on this interesting case. There exist a variety of applications and special algorithmic cases for bipartite graphs [3] that are utilized in such applications as user-product or user-group recommender systems [23, 30], hypergraph based load balancing and mapping [21], gene-disease relationships [4], and drug-to-drug targets [29], to mention just a few.

We define a simple, undirected, and unweighted bipartite graph to be $G = (V, E)$ where the node set is comprised to two *disjoint* subsets, sometimes called "types," $V = A \cup B$. In bipartite graphs edge ij only occurs across types, i.e., $i \in A,\ j \in B$ without loss of generality. A neighborhood of node $i \in V$ is defined as $\Gamma(i) = \{j \in V \mid ij \in E\}$.

We propose two methods for embedding bipartite graphs. These methods fit embeddings by optimizing nodes of each type separately, which we find can lead to higher quality type-specific latent features. Our first method, First-Order Bipartite Embedding (FOBE), samples for the existence of direct, and first-order similarities within the bipartite structure. This approach maintains the separation of types by reformulating $ij \in E$ observations as same-typed observations between i and $\Gamma(j)$, and vice-versa. Our second method, High-Order Bipartite Embedding (HOBE), samples direct, first-, and second-order relationships, and weights samples using algebraic distance on bipartite graphs [6].

[1] Link to anonymized repo. Link to be replaced pending double-blind review.

Preprint. Under review.

Again, we represent sampled relationships between nodes of different types by decomposing them into collections of same-typed relationships. While this sampling approach is similar to FOBE, algebraic distance allows us to improve embedding quality by accounting for broader graph-wide trends. Algebraic distance on bipartite graphs has the effect of capturing strong local similarities between nodes, and reduces the effect of less meaningful relationships. This behavior is beneficial in many applications, such as shopping, where two users are likely more similar if they both purchase a niche hobby product, and may not be similar even if they both purchase a generic cleaning product.

Because FOBE and HOBE each make different prior assumptions about the relevance of bipartite relationships, we propose a method for combining bipartite embeddings to get "best of both worlds" performance. This ensemble approach learns a joint representation from multiple pre-trained embeddings. The "direct" combination method fits a non-linear transformation of the original embeddings into a fixed-size hidden layer in accordance to sampled similarities. The "auto-regularized" combination extends the direct method by introducing a denoising-autoencoder layer in order to regulate the learned joint embedding [28]. The architecture of both approaches maintains a separation between nodes of different types, which allows for type-specific embeddings, without the constraint of a shared global structure. Evaluation of all proposed embeddings is performed on link prediction reinforced with holdout experiments and recommender system tasks.

Our contribution in summary: (1) We introduce First- and High-Order Bipartite Embeddings that learn dense real-valued latent representations of bipartite structure while retaining type-specific semantic information. (2) We present the direct and the auto-regularized methods to leverage multiple pre-trained graph embeddings. This approach can produce a "best of both words" embedding. (3) We discuss the strengths and weaknesses of our proposed methods as they compare to a range of graph embedding techniques. We identify certain graph properties that suit different graph types, and report that none of the proposed embeddings is clearly superior. However, we find that applications wanting to make many same-typed comparisons are often best suited by a type-sensitive embedding.

All code, graphs, and results are publicly available online at: `http://bit.ly/fobe_hobe_code`[2].

Background & Related Work Low-rank embeddings project high-order data into a compressed real-valued space, often for the purpose of facilitating machine learning models. Key advances in text mining by Mikolov et al. [19] leverage the skip-gram model to learn latent semantic features of words from a corpus. Inspired by this approach, Perozzi et al. demonstrate that for a similar method can capture latent structural features of traditional graphs [22]. Their approach, Deepwalk, reduces the graph problem into a text problem by performing a large number of random walks, and then applying the skip-gram model treating each walk as a pseudo-sentence.

An alternative approach, LINE by Tang et al., models first- and second-order node relationships explicitly [27]. Our proposed methods are certainly influenced by LINE's approach, but differ in a few key areas. Firstly, we split our model in order to only make same-typed comparisions. In addition, we introduce terms that compare nodes with relevant neighborhoods, as described in Equation (3). HOBE introduces additional differences by weighting with algebraic distances [6]. Node2Vec blends the intuitions behind both LINE and Deepwalk by combining homophilic and structural similarities through a biased random walk [12].

While the three previously listed embedding approaches are designed for traditional graphs, Metapath2Vec++ by Dong et al. presents a heterogeneous approach using extended skip-gram model [8]. Specifically, this model projects each node into a shared hidden layer, which is then projected to multiple type-specific outputs. Our method differs from Dong et al.'s in a number of ways. Again, we do not apply random walks or the skip-gram model. Furthermore, the Metapath2Vec++ model implicitly asserts that output type-specific embeddings be a linear combination of the same hidden layer. In contrast, we create entirely separate embedding spaces for the nodes of different types.

BiNE by Gao et al. focuses directly on the bipartite case [10]. This approach uses the biased random-walks described in Node2Vec, and samples these walks in proportion to each node's HITS centrality [14]. While our methods differ, again, in the use of skip-gram, BiNE also fundamentally differs from our proposed approaches by enforcing global structure through cross-type similarities.

[2]Link to anonymized repo. Link to be replaced pending double-blind review.

2 Proposed Bipartite Graph Embeddings

We present two sibling strategies for learning bipartite embeddings. First-Order Bipartite Embedding (FOBE) samples direct links from E and first-order relationships between nodes sharing common neighbors. We then fit embeddings to minimize the KL-Divergence between our observations and our embedding-based estimations. The second method, High-Order Bipartite Embedding (HOBE), begins by computing algebraic similarity estimates for each edge [6, 25]. Using these heuristic weights, HOBE samples direct, first- and second-order relationships, to which we fit embeddings using mean-squared error. We implement both methods in Python using Keras [7] and Tensorflow [2].

At a high level, both embedding methods begin by observing structural relationships within a graph. These observations are represented through the functions $\mathbb{S}_X(i,j) : V^2 \to \mathbb{R}$, where $X = \{A, B, AB\}$ represents the sets of nodes observed. Each observed similarity has a corresponding estimated similarity that depends only on embeddings, $\widetilde{\mathbb{S}}_X(\epsilon(i), \epsilon(j))$. Each method must then learn an embedding $\epsilon : V \to \mathbb{R}^k$ to minimize the difference between each $\mathbb{S}_X$ and its corresponding $\widetilde{\mathbb{S}}_X$. However, the specifics of each observation, estimation, and objective differs across methods.

Because we estimate similarities within $\epsilon(A)$ and $\epsilon(B)$ separately, we can better preserve type-specific latent features. This is important for many applications. Consider an embedding of the bipartite graph of viewers and movies, often used for applications such as video recommendations. Within "movie space" one would expect to uncover latent features such as genre, budget, or the presence of high-profile actors. These features are undefined within "viewer space," wherein one would expect to observe latent features corresponding to demographics and viewing preferences. Clearly these two spaces are correlated in a number of ways, such as the alignment between viewer tastes and movie genres. However, we find methods that enforce direct comparisons between viewer and movie embeddings can result in an erosion of type-specific features, which can lead to lower downstream performance. In contrast, the methods proposed here never make a direct assertion of cross-type similarity, and allow implicit relationships to govern any key correlations across spaces.

2.1 First-Order Bipartite Embedding

The goal of FOBE is to model direct and first-order relationships from the original structure. Here, a direct relationship is any edge from the original bipartite graph, while a first-order relationship is defined as $\{(i,j) \mid \Gamma(i) \cap \Gamma(j) \neq \emptyset\}$. Note that nodes in a first-order relationship share the same type. We define observations corresponding with each relationship. Direct observations simply detect the presence of an edge, while first-order relationships similarly detect a common neighbor. Formally:

$$\mathbb{S}_A(i,j) = \mathbb{S}_B(i,j) = \begin{cases} 1 & \Gamma(i) \cap \Gamma(j) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \mathbb{S}_{AB}(i,j) = \begin{cases} 1 & i \in \Gamma(j) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

By sampling γ neighbors, we allow our later embedding model to approximate the effects of Γ, similar to the k-ary set sampling in [20]. Note also that each sample contains one nonzero $\mathbb{S}_X$ value. By fitting all three observations simultaneously, we implicitly generate two negative samples for each positive sample. Furthermore, we generate a fixed number of samples for each node's direct and first-order relationships.

Given these $\mathbb{S}_X$ observations, we fit the ϵ embedding according to corresponding $\widetilde{\mathbb{S}}_X$ estimation functions. To embed a first-order relationship we calculate the sigmoid of the dot product of embeddings (2), namely,

$$\sigma(x) = \frac{1}{1+e^{-x}}. \qquad \text{and} \qquad \widetilde{\mathbb{S}}_A(i,j) = \widetilde{\mathbb{S}}_B(i,j) = \sigma\left(\epsilon(i)^\intercal \epsilon(j)\right) \tag{2}$$

Building from this, we train embeddings based on direct relationships by composing relevant first-order relationships. Specifically, if $ij \in E$ then we would expect i to be similar to $\Gamma(j)$ and vice-versa. Intuitively, a viewer has a higher chance of watching a movie if they are similar to others that have.

We formulate our direct relationship estimate to be the product of each node's average first-order estimate to the other's neighborhood. Formally:

$$\widetilde{\mathbb{S}}_{AB}(i,j) = \mathop{\mathbb{E}}_{k\in\Gamma(j)}\left[\widetilde{\mathbb{S}}_A(i,k)\right] \mathop{\mathbb{E}}_{k\in\Gamma(i)}\left[\widetilde{\mathbb{S}}_B(k,j)\right] \tag{3}$$

In order to train our embedding function ϵ for the FOBE method, we minimize the KL-Divergence [15] between our observed similarities $\mathbb{S}_X$ and our estimated similarities $\widetilde{\mathbb{S}}_X$. We minimize for each simultaneously, for both direct and first-order similarities, using the Adagrad optimizer [9], namely, we solve:

$$\min O_A + O_{AB} + O_B, \qquad \text{and} \qquad O_X = \sum_{i,j\in V} \widetilde{\mathbb{S}}_X(i,j) \log\left(\frac{\mathbb{S}_X(i,j)}{\widetilde{\mathbb{S}}_X(i,j)}\right) \tag{4}$$

2.2 High-Order Bipartite Embedding

Algebraic distance is a measure of dependence between variables popularized in algebraic multigrid (AMG) [24, 5, 18]. Later, it has been shown to be a reliable and fast way to capture implicit similarities between nodes in graphs [13, 17] and hypergraphs that are represented as bipartite graphs [25] (which is leveraged in this paper) taking into account distant neighborhoods. Technically, it is a process of relaxing randomly initialized test vectors using stationary iterative relaxation applied on graph Laplacian homogeneous system of equations, where in the end the algebraic distance between system's variables x_i and x_j (that correspond to linear system's rows i and j) is defined as an maximum absolute value between the ith and jth components of the test vectors (or, depending on application, as sum or sum of squares of them).

In our context, a variable is a node, and we apply K iterations of Jacobi over-relaxation (JOR) on the bipartite graph Laplacian as in [24] ($K = 20$ typically ensures good stabilization as we do not need full convergence, see Theorem 4.2 [6]). Initially, each node's coordinate is assigned a random value, but on each iteration a node's coordinate is updated to move it closer its neighbors' average. Weights corresponding to each neighbor are inversely proportional their degree in order to increase the "pull" of small communities. Intuitively, this acknowledges that two viewers who both watch a niche new-wave movie are more likely similar than two viewers who watched a popular blockbuster. We run JOR on R independent trials (called test vectors in AMG works, convergence proven in [6]). Formally, for rth test vector a_r the update step of JOR is performed as follows, where $a_r^{(t)}(i)$ represents node i's algebraic coordinate on iteration $t \in \{1, .., K\}$, and λ is a damping factor (suggested $\lambda = 0.5$ in [25]).

$$a_r^{(t+1)}(i) = \lambda a_r^{(t)}(i) + (1-\lambda)\frac{\sum_{j\in\Gamma(i)} a_r^{(t)}(j)|\Gamma(j)|^{-1}}{\sum_{j\in\Gamma(i)}|\Gamma(j)|^{-1}} \tag{5}$$

We use the l^2-norm in order to summarize the algebraic distance of two nodes across R trails with different random initializations. As a result, two nodes will be close in our distance calculation if they remain nearby across many trials, which lessens the effect of too slow convergence in a single trial. For our purposes we select $R = 10$. Additionally, we define "algebraic similarity", $s(i,j)$, as a closeness across trials. We subtract the distance between two embeddings from the maximum distance in our space, and rescale the result to the unit interval. Because we know that the maximum distance between any two coordinates in the same trial is 1, we can compute this in constant time:

$$d(i,j) = \sqrt{\sum_{r=1}^{R}\left(a_r^{(K)}(i) - a_r^{(K)}(j)\right)^2} \qquad \text{and} \qquad s(i,j) = \frac{\sqrt{T} - d(i,j)}{\sqrt{T}} \tag{6}$$

After calculating algebraic similarities for pairs of nodes of all edges, we begin to sample direct, first-order, and second-order similarities from the bipartite structure. Here, a second-order connection is one wherein i and j share a neighbor that shares a neighbor: $i \in \Gamma(\Gamma(\Gamma(j)))$. Note that the set of second-order relationships is a superset of the direct relationships. We can extend to these

higher-order connections with HOBE, as opposed to FOBE, because of the information provided in algebraic distances. Many graphs contain a small number of high degree nodes, which creates a very dense second-order graph. Algebraic distances are therefore needed to distinguish which of the sampled second-order connections are meaningful, especially when the refinement is normalized by $|\Gamma(i)|^{-1}$.

We formulate our first-order observations to be equal to the strongest shared bridge between two nodes. This indicates that both nodes are closely related to something that is mutually representative, such as two viewers that watch new-wave cinema. Formally (note that $\mathbb{S}^{'}_{A}$ and $\mathbb{S}^{'}_{B}$ are identical):

$$\mathbb{S}^{'}_{A}(i,j) = \mathbb{S}^{'}_{B}(i,j) = \max_{k \in \Gamma(i) \cap \Gamma(j)} \min\left(s(i,k), s(k,j)\right) \tag{7}$$

When observing second-order relationships, those where i and j are of different types, we again construct a measurement from shared first-order relationships. Specifically, we are looking for the strongest first-order connection between i and j's neighborhood, and vice-versa. In the context of viewers and movies this represents the similarity between a viewer and a movie watched by a friend. Formally:

$$\mathbb{S}^{'}_{AB}(i,j) = \max\left(\max_{k \in \Gamma(j)} \mathbb{S}^{'}_{A}(i,k), \max_{k \in \Gamma(i)} \mathbb{S}^{'}_{B}(k,j)\right) \tag{8}$$

We again collect a fixed number of samples for each relationship type: direct, first- and second-order. We then train embeddings using cosine similarities, however we select the ReLU activation function to replace sigmoid in order to capture the weighted relationships. We optimize for all three observations simultaneously, which again has the effect of creating negative samples for non-observed phenomena. Our estimated similarities are defined as follows:

$$\widetilde{\mathbb{S}}^{'}_{A}(i,j) = \widetilde{\mathbb{S}}^{'}_{B}(i,j) = \max\left(0, \epsilon(i)^{\intercal}\epsilon(j)\right) \tag{9}$$

$$\widetilde{\mathbb{S}}^{'}_{AB}(i,j) = \mathop{\mathbb{E}}_{k \in \Gamma(j)}\left[\widetilde{\mathbb{S}}^{'}_{A}(i,k)\right] \mathop{\mathbb{E}}_{k \in \Gamma(i)}\left[\widetilde{\mathbb{S}}^{'}_{B}(k,j)\right] \tag{10}$$

We use the same model as FOBE to train HOBE, but with our new estimation functions and a new objective. We now optimize for the mean-squared error between our observed and estimated samples, as KL-Divergence is ill-defined for the weighted samples we collect. Formally, we minimize

$$\min O^{\prime}_{A} + O^{\prime}_{AB} + O^{\prime}_{B}, \qquad \text{and} \qquad O^{'}_{X} = \mathop{\mathbb{E}}_{i,j \in V}\left(\mathbb{S}^{'}_{X}(i,j) - \widetilde{\mathbb{S}}^{'}_{X}(i,j)\right)^{2} \tag{11}$$

2.3 Combination Bipartite Embedding

FOBE captures local relationships, while HOBE focuses on higher-order relationships. In order to unify our proposed approaches, we present a method to create a joint embedding from multiple pre-trained bipartite embeddings. This combination method maintains our initial assertion that nodes of different types ought to participate in different global embedding structures.

We fit a non-linear projection of the input embeddings such that an intermediate embedding can accurately uncover direct relationships. This raises a question as to whether it is better to create an intermediate that succeeds in this training task, or whether it is better to fully encode the input embeddings. To address this concern we propose two flavors of our combination method: the "direct" approach maximizes performance on the training task, while the "auto-regularized" approach enforces a full encoding of input embeddings.

The sampling process for both combination approaches is identical. We begin by taking the edge list of the original bipartite graph as our set of positive samples. We then generate five negative samples for each node by selecting random pairs (i,j) such that $i \notin \Gamma(j), i \in A, j \in B$. For each sample, we create an input vector by concatenating each of the m pre-trained embeddings.

$$In(i) = [\epsilon_1(i) \;\; \epsilon_2(i) \;\; ... \;\; \epsilon_m(i)] \tag{12}$$

After generating $In(i)$ and $In(j)$, our model asserts 50% dropout in these input vectors [26]. We do so in the auto-regularized case so that we follow the pattern of denoising auto-encoders, which have shown high performance in robust dimensionality reductions [28]. However, we also find that this dropout increases performance in the direct combination model as well. This is because in either case, we anticipate both redundant and noisy signals to be present across the concatenated embeddings. By adding this dropout factor, we reduce the chance that our combination model will learn to predict edges based on small perturbations between these signals. This is especially necessary for larger values of k and m, where the risk of overfitting increases.

We then project $In(i)$ and $In(j)$ separately onto two hidden layers of size ${}^{d(In)+k'}/_2$ where $d(x)$ indicates the dimensionality of x, and k' represents the desired dimensionality of the combined embeddings. By separating these hidden layers, we only allow signals from within embeddings of the same node to affect its combination. We then project down to two combination embeddings of size k', which act as input to both the joint link-prediction model, as well as to the optional auto-encoder layers.

In the direct case, we simply minimize the mean-squared error between the predicted links and the observed links. Formally, let $\mathbb{S}''(i,j) \to \{0,1\}$ equal the sampled value, and let $\widetilde{\mathbb{S}}''(i,j) \to \mathbb{R}$ be combination estimate. In the auto-regularized case we introduce a factor to enforce that the original (pre-dropout) embeddings can be recovered from the combined embedding. We weight these factors so they are half as important as performing the link prediction training task. If S is the set of samples and $Out(i)$ is the output of the auto-encoder corresponding to $In(i)$, then we optimize the following (direct followed by auto-regularized):

$$\min \mathop{\mathbb{E}}_{i,j \in S} \left(\mathbb{S}''(i,j) - \widetilde{\mathbb{S}}''(i,j) \right)^2 \qquad \text{and} \qquad \min \mathop{\mathbb{E}}_{i,j \in S} \begin{pmatrix} 4 \left(\mathbb{S}''(i,j) - \widetilde{\mathbb{S}}''(i,j) \right)^2 \\ +||In(i) - Out(i)||_2 \\ +||In(j) - Out(j)||_2 \end{pmatrix} \tag{13}$$

3 Link Prediction Experiments

We evaluate the performance of our proposed embeddings across three link prediction tasks and a range of training-test splits. When removing edges, we visit each in random order and remove them with probability h provided the removal does not disconnect the graph. This additional check ensures all nodes appear in all experimental embeddings. The result is the subgraph $G' = (V, E', h)$. Deleted edges form the positive test-set examples, and we generate set of negative samples (edges not present in original graph) of equal size. These samples are used to train three sets of link-prediction models: the A-Personalized, B-Personalized (where A and B are parts of V), and unified models.

A personalized model is a support vector machine trained on the neighborhood of a particular node. A model personalized to $i \in A$ learns to identify a region in B-space corresponding to its neighborhood in G'. We use support vector machines with the radial basis kernel ($C = 1, \gamma = 0.1$) because we find these models result in robust performance given limited training data, and because the chosen kernel function allows for non-spherical decision boundaries. We additionally generate five negative samples for each positive sample (a neighbor of i in G'). In doing so we evaluate the ability to capture type-specific latent features, as each personalized model only considers one-type's embeddings.

The unified link-prediction model, in contrast, learns to associate $ij \in E'$ with a combination of $\epsilon(i)$ and $\epsilon(j)$. This model attempts to quantify global trends across embedding spaces. We use a hidden layer of size k with the ReLU activation function, and a single output with the sigmoid activation. We fit this model against mean-squared error using the Adagrad optimizer [9].

Datasets. We evaluate each embedding across six datasets (see statistics at `http://bit.ly/fobe_hobe_code`). The Amazon, YouTube, DBLP, Friendster, and Livejournal graphs are all taken from the Stanford Large Network Dataset Collection (SNAP) [16]. We select the distribution of each under the listing "Networks with Ground-Truth Communities." Therefore, our bipartite graph consists of users and communities. Furthermore, we collect the MadGrades graph, from an online source provided by the University of Wisconsin at Madison [1]. This graph consists of teachers and course codes, wherein an edge signifies that teacher i has taught course code j at some point. We clean this dataset by iteratively deleting any instructor or course with degree 1 until none remain.

Experimental Parameters. We evaluate the performance of our proposed methods: FOBE and HOBE, as well as our two combination approaches: Direct and Auto-Regularized Combination Bipartite Embedding. We compare against all methods described in Section 1. We evaluate each across the six above graphs and nine training-test splits $h = 0.1, 0.2, ..., 0.9$. Furthermore, we report the performance of A-Personalized, B-Personalized, and unified link prediction models. For all embeddings we select dimensionality $k = 100$. For Deepwalk, we select a walk length of 10, a window size of 5, and 100 walks per node. For LINE we apply the model that combines both first- and second-order relationships, selecting 10,000 samples total and 5 negative samples per node. For Node2Vec we select 10 walks per node, walk length of 7 and a window size of 3. Furthermore, we select typical parameters for BiNE and Metapath2Vec++. For the latter, we supply the metapath of alternating $A - B - A$ nodes, the only metapath in our bipartite case. For FOBE and HOBE we generate 200 samples per node, and when sampling neighborhoods we select 5 nodes with replacement upon each observation. After training both methods, we fit the Direct and Auto-Regularized Combination methods, each trained using *only* the results of FOBE and HOBE.

Sensitivity Study. We select the MadGrades network to demonstrate how our proposed methods are effected by the sampling rate. We run ten trials for each experimental sampling rate, consisting of powers of 2 from 1 to 1024. Each trial represents an independent 50% holdout experiment. We present min, mean, and max observed link prediction accuracy.

4 Recommendation Experiments

We follow the procedure originally described by Gao et al. and evaluate our proposed embeddings through the task of recommendation [10]. Recommendation systems propose products to users in order to maximize the overall interaction rate. These systems fit the bipartite graph model because they are defined on the set of user-product interactions. While there are many similarities between recommendation and link prediction, the key difference is the introduction of weighted connections. As a result, recommendation systems are evaluated based on their ability to rank products in accordance to held-out user supplied rankings. This is quantified through a number of metrics defined on the top k system-supplied recommendation for each user. When using embeddings to make a comparison, Gao et al. rank products by their embedding's dot product with a given user. However, our proposed methods relax the constraint that products and users be directly comparable. As a result, when ranking products for a particular user for our proposed embeddings we must first define a product-space representation. For each user we collect the set of known product ratings, and calculate a product centroid weighted by those ratings.

Experimental Procedure. We present a comparison between our proposed methods and all previously discussed embeddings across the DBLP[3] and LastFM[4] datasets. Note that this distribution of DBLP is the bipartite graph of authors and venues, and is different from the community-based version distributed by SNAP. The LastFM dataset consists of listeners and musicians, where an edge indicates listen count, which we log-scale to improve convergence for all methods. We start by splitting each rating set into training- and test-sets with a 40% holdout. In the case of DBLP we use the same split as Gao et al. We use embeddings from the training bipartite graph to perform link prediction. We then compare the ranked list of training-set recommendations for each user, truncated to 10 items, to the test-set rankings. We calculate 128-dimensional embeddings for each method, and report F1, Normalized Discounted Cumulative Gain (NDCG), Mean Average Precision (MAP) and Mean Reciprocal Rate (MRR).

5 Results and Discussion

In contrast to what is typically claimed in papers, we observe that the link prediction data (Table 1) demonstrates that different graphs lead to very different performance results for the existing state-ot-the-art and proposed embeddings. Moreover, their behavior is changed with different holdouts when the size of training set is smaller. For instance, our methods are above the state of the art in the Youtube and MadGrades graphs, but Metapath2Vec++, Node2Vec, and LINE each have scenarios wherein they outperform the field. Additionally, while there are scenarios where the combination

[3] `https://github.com/clhchtcjj/BiNE/tree/master/data/dblp`

[4] `https://grouplens.org/datasets/hetrec-2011/`

methods perform as expected, such as in the Youtube, MadGrades, and DBLP B-Personalized cases, we observe that variability in the other proposed embeddings can disrupt this performance gain. Looking to the sensitivity study (Tables 2), we see the variability of HOBE is significantly larger for small sampling rates. However, we do observe that after approximately 32 samples per node, in the case of MadGrades, this effect is reduced. Still, considering FOBE does not exhibit this same quality, it is likely the variability of the algebraic similarity measure that ultimately leads to otherwise unexpected reductions in HOBES performance.

In the recommendation results (Table 3 and 4), our methods clearly improve the state-of-the art across multiple metrics. This is further evidence that our sampling decompositions are better able to capture product-specific features. Specifically, our biggest increase is in MRR for DBLP, which indicates that the first few suggestions from our embeddings are often more relevant. This is best seen with HOBE, demonstrating the ability for algebraic distance to estimate useful local similarity measures. While we note some improvement in the LastFM dataset, the effect is not as significant, and FOBE outperforms HOBE. One reason for this is that LastFM contains significantly more artists-to-user than DBLP contains venues-to-author. As a result the amount of information present when estimating algebraic similarities is different across datasets, and insufficient to boost HOBE above FOBE.

To continue comparing FOBE and HOBE, it would appear that higher-order sampling is often able to produce better results, but that the algebraic distance heuristic introduces added variability that occasionally reduces overall performance. In some applications it would appear that this variability is manageable, as seen in our DBLP recommendation results. However in the case of link prediction on Amazon communities, this caused an unintentional drop when FOBE remained more consistent. Overall, FOBE and HOBE are fast methods that broaden the array of embedding techniques available for bipartite graphs. While no method is clearly superior in every case, there exist a range of graphs and applications that are better suited by these methods.

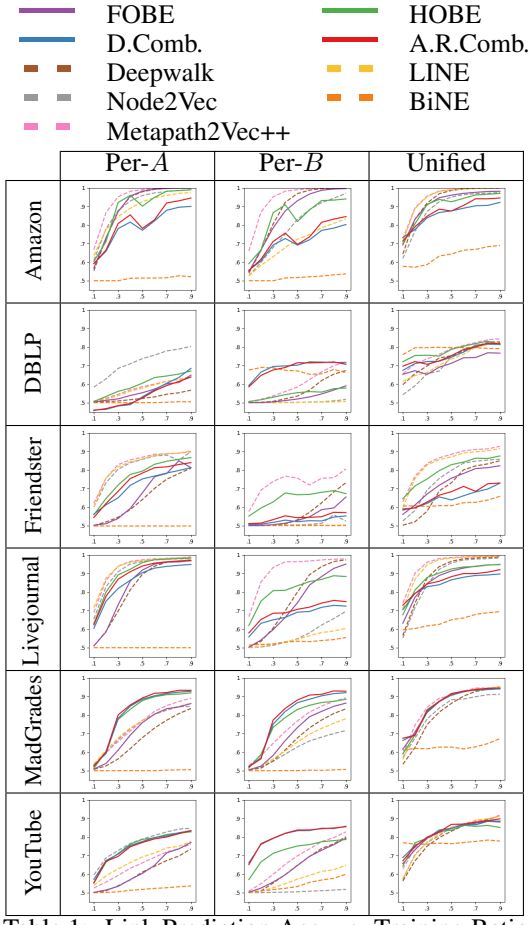

Table 1: Link Prediction Acc. vs. Training Ratio.

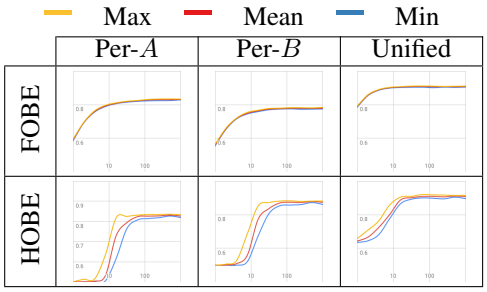

Table 2: Link Pred. Acc. vs. Sampling Rate.

Metric@10:	F1	NDCG	MAP	MRR
DeepWalk	.0850	.2414	.1971	.3153
LINE	.0899	.1441	.0962	.1713
Node2Vec	.0854	.2389	.1944	.3111
MP2V++	.0865	.2514	.1906	.3197
BINE	**.1137**	.2619	.2047	.3336
FOBE	.1108	.3771	.2382	.4491
HOBE	.1003	**.4054**	**.3156**	**.6276**
D.Comb.	.0753	.2973	.2362	.5996
A.R.Comb.	.0667	.2359	.1730	.5080

Table 3: DBLP Recommendation. Note: result numbers from prior works are reproduced from [10].

Metric@10:	F1	NDCG	MAP	MRR
DeepWalk	.0027	.0153	.0069	.1844
LINE	.0067	.0435	.0229	.2477
Node2Vec	.0279	.1261	.0645	.2047
MP2V++	.0024	.0153	.0088	.2677
BINE	.0227	.1551	.0982	.3539
FOBE	**.0729**	**.3085**	**.1997**	.3778
HOBE	.0195	.1352	.0789	.3400
D.Comb.	.0243	.1285	.0795	.3520
A.R.Comb.	.0388	.1927	.1249	**.3915**

Table 4: LastFM Recommendations.

References

[1] MadGrades - UW Madison Grade Distributions. `https://madgrades.com`. Accessed: 2018-10-25.

[2] ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., ET AL. Tensorflow: a system for large-scale machine learning. In *OSDI* (2016), vol. 16, pp. 265–283.

[3] ASRATIAN, A. S., DENLEY, T. M., AND HÄGGKVIST, R. *Bipartite graphs and their applications*, vol. 131. Cambridge university press, 1998.

[4] BARABÁSI, A.-L., GULBAHCE, N., AND LOSCALZO, J. Network medicine: a network-based approach to human disease. *Nature reviews genetics 12*, 1 (2011), 56.

[5] BRANDT, A., BRANNICK, J. J., KAHL, K., AND LIVSHITS, I. Bootstrap AMG. *SIAM J. Scientific Computing 33*, 2 (2011), 612–632.

[6] CHEN, J., AND SAFRO, I. Algebraic distance on graphs. *SIAM Journal on Scientific Computing 33*, 6 (2011), 3468–3490.

[7] CHOLLET, F., ET AL. Keras. `https://keras.io`, 2015.

[8] DONG, Y., CHAWLA, N. V., AND SWAMI, A. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017), ACM, pp. 135–144.

[9] DUCHI, J., HAZAN, E., AND SINGER, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research 12*, Jul (2011), 2121–2159.

[10] GAO, M., CHEN, L., HE, X., AND ZHOU, A. BiNE: Bipartite Network Embedding. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (New York, NY, USA, 2018), SIGIR '18, ACM, pp. 715–724.

[11] GOYAL, P., AND FERRARA, E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems 151* (2018), 78–94.

[12] 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* (2016), ACM, pp. 855–864.

[13] JOHN, E., AND SAFRO, I. Single-and multi-level network sparsification by algebraic distance. *Journal of Complex Networks 5*, 3 (2016), 352–388.

[14] KLEINBERG, J. M. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM) 46*, 5 (1999), 604–632.

[15] KULLBACK, S., AND LEIBLER, R. A. On information and sufficiency. *The annals of mathematical statistics 22*, 1 (1951), 79–86.

[16] LESKOVEC, J., AND KREVL, A. {SNAP Datasets}:{Stanford} large network dataset collection.

[17] LEYFFER, S., AND SAFRO, I. Fast response to infection spread and cyber attacks on large-scale networks. *Journal of Complex Networks 1*, 2 (2013), 183–199.

[18] LIVNE, O. E., AND BRANDT, A. Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver. *SIAM Journal on Scientific Computing 34*, 4 (2012), B499–B522.

[19] MIKOLOV, T., CHEN, K., CORRADO, G., AND DEAN, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[20] MURPHY, R. L., SRINIVASAN, B., RAO, V., AND RIBEIRO, B. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *International Conference on Learning Representations* (2019).

[21] NAUMANN, U., AND SCHENK, O. *Combinatorial scientific computing*. CRC Press, 2012.

[22] 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* (2014), ACM, pp. 701–710.

[23] PERUGINI, S., GONÇALVES, M. A., AND FOX, E. A. Recommender systems research: A connection-centric survey. *Journal of Intelligent Information Systems 23*, 2 (2004), 107–143.

[24] RON, D., SAFRO, I., AND BRANDT, A. Relaxation-based coarsening and multiscale graph organization. *Multiscale Modeling & Simulation 9*, 1 (2011), 407–423.

[25] SHAYDULIN, R., CHEN, J., AND SAFRO, I. Relaxation-based coarsening for multilevel hypergraph partitioning. *SIAM Multiscale Modeling and Simulation 17* (2019), 482–506.

[26] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research 15*, 1 (2014), 1929–1958.

[27] 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* (2015), International World Wide Web Conferences Steering Committee, pp. 1067–1077.

[28] VINCENT, P., LAROCHELLE, H., BENGIO, Y., AND MANZAGOL, P.-A. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 1096–1103.

[29] YILDIRIM, M. A., GOH, K.-I., CUSICK, M. E., BARABÁSI, A.-L., AND VIDAL, M. Drug—target network. *Nature biotechnology 25*, 10 (2007), 1119.

[30] ZHANG, C., HU, S., TANG, Z. G., AND CHAN, T.-H. H. Re-revisiting learning on hypergraphs: Confidence interval and subgradient method. In *Proceedings of the 34th International Conference on Machine Learning* (International Convention Centre, Sydney, Australia, 06–11 Aug 2017), D. Precup and Y. W. Teh, Eds., vol. 70 of *Proceedings of Machine Learning Research*, PMLR, pp. 4026–4034.