BiANE: Bipartite Attributed Network Embedding

Wentao Huang[1*], Yuchen Li[2,3], Yuan Fang[2], Ju Fan[1†], Hongxia Yang[4]
[1]Renmin University of China, [2]Singapore Management University, [3]Zhejiang University, [4]Alibaba Group
{wentaohuang,fanj}@ruc.edu.cn,{yuchenli,yfang}@smu.edu.sg,yang.yhx@alibaba-inc.com

ABSTRACT

Network embedding effectively transforms complex network data into a low-dimensional vector space and has shown great performance in many real-world scenarios, such as link prediction, node classification, and similarity search. A plethora of methods have been proposed to learn node representations and achieve encouraging results. Nevertheless, little attention has been paid on the embedding technique for bipartite attributed networks, which is a typical data structure for modeling nodes from two distinct partitions.

In this paper, we propose a novel model called BiANE, short for *Bipartite Attributed Network Embedding*. In particular, BiANE not only models the inter-partition proximity but also models the intra-partition proximity. To effectively preserve the intra-partition proximity, we jointly model the attribute proximity and the structure proximity through a novel latent correlation training approach. Furthermore, we propose a dynamic *positive* sampling technique to overcome the efficiency drawbacks of the existing dynamic *negative* sampling techniques. Extensive experiments have been conducted on several real-world networks, and the results demonstrate that our proposed approach can significantly outperform state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Dimensionality reduction and manifold learning**; • **Information systems** → *Social networks*.

KEYWORDS

Network embedding; Bipartite attributed network; Link prediction

ACM Reference Format:
Wentao Huang[1*], Yuchen Li[2,3], Yuan Fang[2], Ju Fan[1†], Hongxia Yang[4]. 2020. BiANE: Bipartite Attributed Network Embedding. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401068

*Work done during an internship at Advanced Digital Sciences Center, Singapore.
†Ju Fan is the corresponding author.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401068

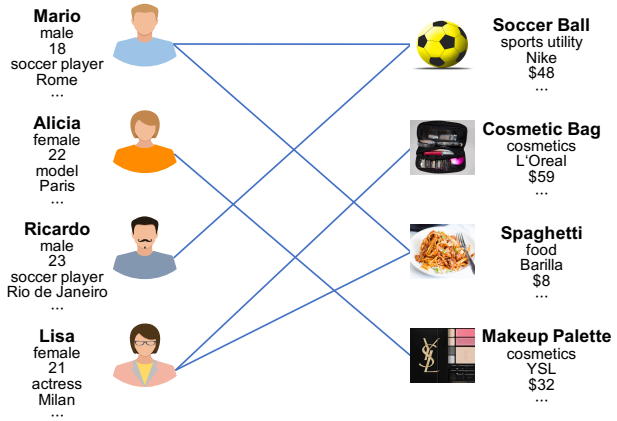


Figure 1: An e-commerce example of bipartite attributed networks. Nodes in the network can be categorized into two attributed partitions: users and items.

1 INTRODUCTION

Networks are ubiquitous data structures that have been extensively utilized to model a plethora of real-world problems such as molecular structures [6], social networks [22–24, 38], and recommender systems [41, 43]. Network embedding has become an efficient and effective approach to model network structure and has attracted significant attention recently. It aims to transform a network into a low-dimensional vector space and has shown great performance in many downstream tasks, such as link prediction, node classification, and similarity search.

This paper studies the *bipartite attributed networks*, which are not well addressed by existing efforts on network embedding. In a bipartite attributed network, nodes are categorized into two *partitions* and edges only exist between nodes from different partitions. Moreover, each node is associated with a set of attributes. Bipartite attributed networks can model many real-world scenarios. For example, as illustrated in Figure 1, user activities on an e-commerce website can be formalized as a bipartite attributed network: nodes are categorized into users and items, while edges among nodes represent the interactive behaviors between users and items. Nodes may have attributes, such as user profiles (e.g., gender, age, occupation, location) and item features (e.g., category, brand, price). Other typical applications include bibliographical networks [2], biological networks [30], and risk assessment [17].

In principle, there are two key observations that lie at the core of bipartite attributed network embedding. First, we need to preserve the *inter-partition proximity* among the nodes, which captures the across-partition relationships conveyed by the edges of bipartite networks. More importantly, we also need to preserve the *intra-partition proximity* that refers to the underlying proximity within each partition. To be more specific, the intra-partition proximity

consists of both *structure* and *attribute* aspects for nodes within the same partition. On the one hand, nodes, which have similar "interaction" behaviors with the nodes in the other partition, should have high proximity. For example, if two users co-like the same items (e.g., users Mario and Ricardo), they should have larger potential high-order similarity. Previous works [10, 11, 18, 26, 43, 46] have claimed this point and validated that the modeling such high-order similarity can enhance the performance of recommendation tasks. On the other hand, each node is associated with a set of attributes, which can provide auxiliary information to enhance representation learning. The intuition is that similar nodes tend to share similar attributes, resulting to similar behaviors in the network. Take Figure 1 again as an example. Alicia and Lisa are both young ladies, and they probably would spend some money consuming cosmetics.

Despite numerous solutions have been proposed for network embedding recently, we argue that they are not sufficient for embedding bipartite attributed networks as they fail to effectively preserve the intra-partition proximity for following reasons.

First, previous works overlook the specific "correlation" between the attribute and structure proximity when preserving them. The attribute information and the structure information are two different modalities, they have a highly non-linear relationship. However, since they describe different aspects of the same node, they not only present mutual complementarity but also share an underlying coherence in the embedding manifold, to which we refer as *the attribute-structure correlation*. For instance, two scholars with similar research interests (attribute) may co-publish papers (structure), or both have some connections with specific researchers (structure). Unfortunately, existing methods neither preserve the attribute proximity and the structure proximity simultaneously nor model the attribute-structure correlation effectively due to the highly non-linear relationship. Thus they render inferior embeddings.

Second, modeling the high-order structure proximity within each partition introduces scalability issues. Existing works [10, 11] add large amounts of additional links within each partition to benefit from modeling such structure proximity, but they also make the network much denser than before and introduce the scalability problem. The scalability problem is even more exaggerated when it comes to the case of embedding methods with dynamic negative sampling strategies, which introduce severe computation complexities. A common solution is to use static negative sampling strategies to reduce complexity. Nevertheless, previous works [8, 9, 48, 49] have proved that static negative sampling strategies fail to reflect the varying node information distribution in the embedding space during the training process and may lead to vanishing gradients (see Section 2.3 for more details). Hence, they suffer from scalability issues and yield unsatisfactory results on large networks.

Motivated by the aforementioned gaps, we propose a novel embedding model, namely *Bipartite Attributed Network Embedding*, or BiANE for short. In order to model the intra-partition proximity, we augment intra-partition edges to existing edges presented in the network for intra-partition modeling. Then we employ autoencoders to compress the attribute information and the structure information separately and specifically devise a strategy called *Latent Correlation Training* by introducing auxiliary transforming kernels to project the attribute and structure encodings to a novel latent space to model the attribute-structure correlation. Meanwhile, we perform inter-partition modeling to tune model parameters and ameliorate the learning embeddings. Furthermore, so as to obtain satisfying embeddings and overcome complexity defects caused by dynamic *negative* sampling strategies, we propose a novel dynamic *positive* sampling technique and draw samples from a k-nearest neighbor index to train the model effectively and efficiently.

To summarize, we make the following contributions.

(1) We study the problem of bipartite attributed network embedding and propose a novel model called BiANE. BiANE extracts the attribute information as well as the structure information and effectively aggregates them together and maintain their underlying correlation. BiANE simultaneously models the inter-partition proximity and the intra-partition proximity of the network and integrates each other to ameliorate the learning result.

(2) We introduce an efficient dynamic *positive* sampling strategy, which improves the training result and makes our model scalable for large scale networks.

(3) Extensive empirical studies are conducted on several real-world network datasets and experimental results demonstrate the superior efficacy and efficiency of the proposed BiANE model.

2 RELATED WORK

In this section, we review some existing works related to our work and address their drawbacks for bipartite attributed network embedding. To save space, we only name a few state-of-the-art works.

2.1 Network Embedding

Network embedding has become a hot topic across multiple fields. Earlier works DeepWalk [29] and node2vec [13] utilize random walks to simulate word sequences and then apply a Skip-gram model [28] to learn node embeddings. Metapath2vec [5] and HIN2vec [7] extends DeepWalk [29] by exploiting different types of metapaths to guide random walks on heterogenous networks. HEP [51] uses a neighborhood propagation mechanism for embedding reconstruction. ANRL [50] adopts neighborhood enhancement autoencoders and attribute-aware skip-gram models to preserve the attribute proximity and the structure proximity. AANE [15] is fabricated in a distributed manner and utilizes a joint learning process for attributed network embedding. However, they neither preserve the attribute proximity nor model the attribute-structure correlation explicitly. Hence, they are inadequate for embedding bipartite attributed networks.

2.2 Bipartite Network Modeling

Most efforts on bipartite networks are exerted on node ranking or click-through rate (CTR) problems. BPR [31] and RankALS [33] rank nodes by optimizing pairwise-ranking objectives. Wide&Deep [4] and DeepFM [14] propose deep models to tackle CTR problems. However, all these works only output a single score, which fails to convey rich information like network embeddings.

BiNE [10] is the first work that is designed to embedding bipartite networks. It specifically models the structure proximity within each partition by augmenting intra-partition edges. NGCF [43] produces node embedding by modeling the collaborative signal and high-order connectivities of user-item bipartite interactions. GC-MC [36] and STAR-GCN [47] employ graph convolution networks

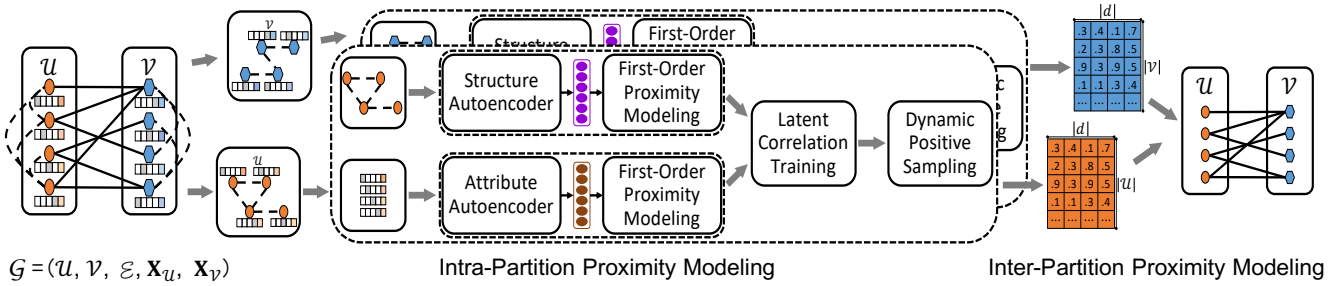


Figure 2: The overall framework of BiANE model. The learning algorithm can be mainly divided into two phases: (1) Intra-Partition Proximity Modeling, where we address two key points in training: *Latent Correlation Training* and *Dynamic Positive Sampling* (2) Inter-Partition Proximity Modeling. The whole model is trained in an end-to-end manner. Dashed lines in $\mathcal{G}$ represent underlying intra-partition neighbor relationships within each partition.

[20] to perform matrix completion on bipartite networks. Nevertheless, they either ignore the node attributes or are not scalable in modeling the attribute-structure correlation. Therefore, they yield unsatisfactory results for bipartite attributed network embedding.

2.3 Negative Sampling Strategy

Another line of related works is about negative sampling. Negative sampling is a critical step in embedding learning as it guides the direction of gradient variation. Most models [1, 29, 34] adopt a pre-defined static distribution for negative sampling, e.g., uniform distribution, Bernouli distribution or a distribution that is proportional to $\frac{3}{4}$ power of the node degree. As the training proceeds, the static distribution can not capture the variation of embedding space. Static sampling may also introduces the vanishing gradient problem in the later training phase [3, 42, 49].

Other models [8, 9, 48, 49] resort to a dynamic sampling strategy to circumvent the static sampling problem, but they suffer from high computation complexity ($O(n^2)$). GraphGAN [39] tries to reduce the complexity by constructing BFS-trees in the offline stage. Unfortunately, the offline construction phase still requires a quadratic complexity. Thus, such an approach can only handle graphs with at most a few thousands of nodes.

3 METHODOLOGY

In this section, we propose BiANE, an end-to-end model for bipartite attributed networks representation learning (see Figure 2 for an overview). We first illustrate the notations and present the definition of the bipartite attributed network embedding problem. The overall framework of our BiANE model will be introduced subsequently. Next, we elaborate on our model in detail. At last, we give some discussions and analysis on the proposed model.

3.1 Notations and Problem Definition

Throughout this paper, matrices are denoted as uppercase bold letters (e.g., $\mathbf{A}$), vectors as lowercase bold letters (e.g., $\mathbf{a}$), and scalars are denoted as lowercase italic alphabets (e.g., p). The transpose of a matrix or a vector is denoted as $\mathbf{A}^\mathrm{T}$ or $\mathbf{a}^\mathrm{T}$. We describe the i^{th} row of matrix $\mathbf{A}$ as $\mathbf{a}_i$ and the element of i^{th} row and j^{th} column as a_{ij}. Key notations are summarized in the Table 1.

Let $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E}, \mathbf{X}_\mathcal{U}, \mathbf{X}_\mathcal{V})$ be a bipartite attributed network, where $\mathcal{U}$ denotes the node set for one type while $\mathcal{V}$ denotes node set for the other. $\mathcal{E}$ stands for the edge set of the network. e_{mn} denotes the an edge in $\mathcal{E}$ between two nodes u_m and v_n where $u_m \in \mathcal{U}$ and $v_n \in \mathcal{V}$. $\mathbf{X}_\mathcal{U}$ and $\mathbf{X}_\mathcal{V}$ stand for the attribute information matrix for $\mathcal{U}, \mathcal{V}$ respectively. The goal of bipartite attributed network embedding is to map all nodes in $\mathcal{G}$ to a vector in the low-dimensional embedding space $\mathbf{H} \in \mathbb{R}^d$, where each node is represented as a d-dimension embedding vector $\mathbf{h}$.

Table 1: Notations

Notations	Definition
$\mathbf{W}_x^{(k)}, \mathbf{W}_z^{(k)}$	encoder parameters
$\widehat{\mathbf{W}}_x^{(k)}, \widehat{\mathbf{W}}_z^{(k)}$	decoder parameters
$\widetilde{\mathbf{W}}_x^{(k)}, \widetilde{\mathbf{W}}_z^{(k)}$	transforming kernel parameters
$\mathbf{x}$, $\mathbf{z}$	attribute and high-order structure features
$\hat{\mathbf{x}}$, $\hat{\mathbf{z}}$	reconstructed vectors
$\mathbf{x}'$, $\mathbf{z}'$	autoencoder encodings
$\tilde{\mathbf{x}}$, $\tilde{\mathbf{z}}$	latent representations
$\mathbf{h}$	final embeddings

We present some key concepts as follows:

Concept 1. *(The Attribute Proximity).* Given any two nodes from a bipartite attributed network, the attribute proximity measures the similarity of their attributes. Nodes with similar attributes should have their embedding vectors closer in the latent space.

Concept 2. *(The Structure Proximity)* The goal is to capture the topological similarity from the graph structure. Specifically, the structure proximity can be interpreted in two aspects: the first-order proximity and the high-order proximity. The first-order proximity is determined by if there is a direct linking edge between two nodes. The high-order proximity represents the similarity between the neighbor sets of two nodes.

3.2 Overall Framework

The overall framework of BiANE is shown in Figure 2. The framework preserves two kinds of proximity: the inter-partition proximity and the intra-partition proximity. Given a bipartite attributed network, we first perform partitioning on the network according to different node types. Subsequently, we extract the structure information and the attribute information from either partition and feed

them into different autoencoders. We incorporate the first-order proximity preserving in the autoencoder learning process and propose a novel attribute-structure correlation training in another latent space for each partition. At last, we aggregate the encoding vectors from the aforementioned autoencoders and model the inter-partition proximity of the bipartite attributed network.

3.3 Inter-Partition Proximity Modeling

For bipartite attributed networks, the most important properties are the links that exist across two partitions, to which we refer as the inter-partition relationships of bipartite networks. For the nodes that are linked across partitions, we must ensure that their embeddings should capture the inter-partition relationship in the embedding space. For an edge e_{mn} between u_m and v_n, we regard their joint probability as the inter-partition proximity between them:

$$p(m,n) = \sigma(\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_n), \tag{1}$$

where σ stands for the sigmoid function. Here we treat $\mathbf{h}_m$ and $\mathbf{h}_n$ as the final embedding for node u_m and node v_n respectively. We will explain how we get the embedding vector $\mathbf{h}$ in the following subsection. We maximize the likelihood function of joint probability by minimizing the following loss function:

$$L_1 = -\sum_{e_{mn}\in\mathcal{E}} \log\sigma(\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_n) - \sum_{n'=1} \mathbb{E}_{v_{n'}\sim P_n(v)} \log\sigma(-\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_{n'}), \tag{2}$$

where $P_n(v)$ is the negative sampling distribution.

3.4 Intra-Partition Proximity Modeling

We now illustrate how to integrate the intra-partition proximity modeling with the inter-partition proximity modeling. Without loss of generality, we use $\mathcal{U}$ to elaborate and the same idea goes for the partition $\mathcal{V}$.

BiANE captures both the attribute proximity and the structure proximity for intra-partition modeling. For each node u, we first extract raw features for the attribute information and the structure information as $\mathbf{x}$ and $\mathbf{z}$, respectively. Subsequently, $\mathbf{x}$ and $\mathbf{z}$ are fed into two independent autoencoders to learn their compact representations $\mathbf{x}'$ and $\mathbf{z}'$. Finally, we jointly model the attribute proximity and the structure proximity in a single loss function.

Raw Feature Extraction: Each node is associated with attributes. We normalize the numerical attributes and use one-hot to encode categorical attributes. After that, we concatenate all of them as our attribute information $\mathbf{x}$. To obtain the structure information for modeling the intra-partition proximity, we connect two nodes u_m and u_n from $\mathcal{U}$ if they share a common neighbor $v \in \mathcal{V}$ and the connected edge could be assigned a weight (e.g., Jaccard or cosine similarity on their neighbors in $\mathcal{V}$). The directly connected nodes are referred as the *intra-partition neighbors*. Based on the intra-partition neighbors, we can synthesize an intra-partition network within each partition. The intra-partition neighbors represent the first-order relationships for nodes within the same partition. To model the high-order structure proximity, we build the high-order structure proximity matrix $\mathbf{Z}$ as the following:

$$\mathbf{Z} = \mathbf{A}^{\mathbf{1}} + \mathbf{A}^{\mathbf{2}} + \cdots + \mathbf{A}^{\mathbf{k-1}} + \mathbf{A}^{\mathbf{k}}, \tag{3}$$

where $\mathbf{A}^k$ is the normalized k-step probability transition matrix on the intra-partition network and $z_{ij} \in \mathbf{Z}$ denotes the high-order proximity feature of i^{th} node and j^{th} node. By this means, we can obtain the high-order structure information $\mathbf{z}$ for each node inside the partition.

Compact Feature Learning: We feed the raw features extracted into two separate autoencoders for the attribute information and the structure information respectively. See Figure 3 for illustration. We feed the features $\mathbf{x}$ and $\mathbf{z}$ into two independent autoencoders to obtain encodings $\mathbf{x}'$ and $\mathbf{z}'$ as well as reconstructed vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$.

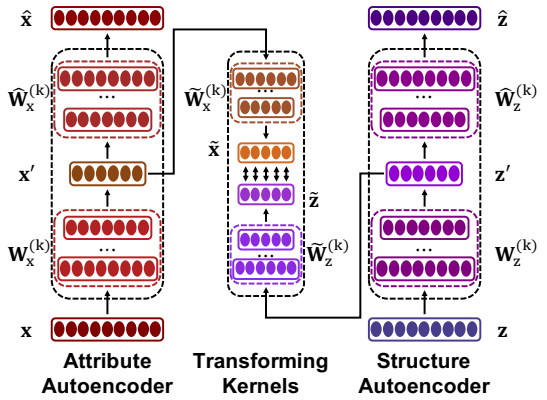


Figure 3: The architecture for the intra-partition proximity modeling module. Specifically, *Latent Correlation Training* and *Dynamic Positive Sampling* are performed in the latent space after we transform $\mathbf{x}'$ and $\mathbf{z}'$ to $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$ respectively by transforming kernels.

We minimize the following reconstruction loss function to capture the attribute information and the structure information:

$$L_2 = \sum_i \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|^2 + \sum_i \|\hat{\mathbf{z}}_i - \mathbf{z}_i\|^2. \tag{4}$$

With this approach, both the attribute information and the structure information can be effectively extracted as the reconstruction process of autoencoders can enhance neural components to capture the data manifold smoothly and reduce noise [32].

Joint Modeling: We jointly model the attribute encoding and the structure encoding to preserve the first-order proximity by optimizing the following loss:

$$\begin{aligned} L_3 = &-\sum_{a_{mn}>0} \log\sigma({\mathbf{x}'_m}^{\mathrm{T}} \cdot \mathbf{x}'_n) - \sum_{n'=1} \mathbb{E}_{v_{n'}\sim P'_n(v)} \log\sigma(-{\mathbf{x}'_m}^{\mathrm{T}} \cdot \mathbf{x}'_{n'}) \\ &-\sum_{a_{mn}>0} \log\sigma({\mathbf{z}'_m}^{\mathrm{T}} \cdot \mathbf{z}'_n) - \sum_{n'=1} \mathbb{E}_{v_{n'}\sim P'_n(v)} \log\sigma(-{\mathbf{z}'_m}^{\mathrm{T}} \cdot \mathbf{z}'_{n'}), \end{aligned} \tag{5}$$

where a_{mn} denotes the element of the adjacency matrix for the synthesized intra-partition network and $P'_n(v)$ is the negative sampling distribution.

Finally, we concatenate $\mathbf{x}'$ and $\mathbf{z}'$ to obtain the final embedding $\mathbf{h}$ and utilize $\mathbf{h}$ for the inter-partition proximity modeling.

3.5 Latent Correlation Training

The attribute information and the structure information are two different modalities, they provide complementary information for each

other. Besides, they both describe the same network, which implies they share an underlying coherence. As mentioned in Section 1, we consider their coherence and complementarity together and generally name it as *the attribute-structure correlation*. To achieve the maximum utilization of the attribute information and the structure information, we should incorporate such correlation in network embeddings as much as possible.

Nevertheless, these two information reflect different properties of the network, their latent modalities are highly non-linear, introducing great difficulties in modeling the attribute-structure correlation. [8] has proved that neither shallow operations (e.g., concatenation and element-wise difference) nor sharing neural layers can be sufficient to capture the mutual coherence or complementarity.

To address the issue, we introduce a novel training procedure to preserve the attribute-structure correlation. Considering $\mathbf{x}'$ and $\mathbf{z}'$ in encoding space, an intuitive approach is to enhance their joint probability to enforce them coherently. Besides, since they are extracted from separate autoencoders, they deliver complementary information. However, since $\mathbf{x}'$ and $\mathbf{z}'$ have been utilized to model the first-order proximity in their own encoding space directly, adding extra loss functions to optimize their joint probability may introduce additional disturbances in their encoding space, making the aforementioned joint modeling training more unstable and affect the overall training in a non-deterministic manner.

To overcome such limitations, we employ two auxiliary space transforming kernels to switch the space from encodings to a new latent space and project the encodings to latent representations (see Figure 3 for illustration):

$$\tilde{\mathbf{x}} = \delta^{(k)}(\widetilde{\mathbf{W}}_x^{(k)}(\cdots \delta^{(1)}(\widetilde{\mathbf{W}}_x^{(1)}\mathbf{x}' + \tilde{\mathbf{b}}_x^{(1)})\cdots) + \tilde{\mathbf{b}}_x^{(k)}), \tag{6a}$$

$$\tilde{\mathbf{z}} = \delta^{(k)}(\widetilde{\mathbf{W}}_z^{(k)}(\cdots \delta^{(1)}(\widetilde{\mathbf{W}}_z^{(1)}\mathbf{z}' + \tilde{\mathbf{b}}_z^{(1)})\cdots) + \tilde{\mathbf{b}}_z^{(k)}). \tag{6b}$$

We define the correlation between the attribute information and the structure information for any two nodes as the joint probability of their latent representations:

$$\tilde{p}(m,n) = \sigma(\tilde{\mathbf{x}}_m^{\mathrm{T}} \cdot \tilde{\mathbf{z}}_n). \tag{7}$$

If two nodes are linked in the synthetic intra-partition network, it is very likely to infer that their attribute information and structure information have a strong correlation. We maximize the aforementioned joint probability in the latent space to ensure the correlation can be incorporated in our model. For each node u, the node with the strongest correlation is u itself. Thus, we should maximize the joint probability of its own $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$ in the new latent space:

$$L_4 = -\sum_{m=n} \log\sigma(\tilde{\mathbf{x}}_m^{\mathrm{T}} \cdot \tilde{\mathbf{z}}_n) - \sum_{n'=1} \mathbb{E}_{v_{n'}\sim P'_n(v)} \log\sigma(-\tilde{\mathbf{x}}_m{}^{\mathrm{T}} \cdot \tilde{\mathbf{z}}_{n'}), \tag{8}$$

where $P'_n(v)$ is the negative sampling strategy. By this means, we can ameliorate the parameters of encoders and transforming kernels through model training and enforce the attribute-structure correlation is well integrated into our model parameters.

3.6 Dynamic Positive Sampling Strategy

Negative sampling plays an important role in the network embedding optimization process, which requires bunches of negative cases to help determine the boundary of "similarity". Traditional approaches adopt a static sampling strategy, e.g., random sampling or popularity sampling. However, static approaches keep the sampling criterion unchanged during training, which does not reflect the node informativeness in the varying embedding space. Moreover, static sampling provides the same negative distribution for different nodes, which fail to capture the local features of the network, resulting in the gradient vanishing issue or a poor local optimum problem for training [3, 42, 49].

One possible solution is to sample negative cases from a dynamic negative sampling distribution [8, 9]. To be more specific, for each node, we calculate the joint probabilities from a node u to all other nodes. Then, the negative samples for u are generated by following a distribution that is proportional to the dynamically calculated joint probabilities. Due to dynamic sampling, the distribution can adapt to the training processing and is able to circumvent the potential vanishing gradient problem [9, 49], yielding satisfactory results. However, this approach costs $O(n^2)$ complexity for dynamic distribution computation in each round, which will severely degenerate the scalability and efficiency of the training process. Such deficiency is even exaggerated in our scenario since we physically augment intra-partition links for modeling the structure proximity within the partition, which significantly "enlarges" the network, making it too dense to update the sampling distribution dynamically.

Dynamic Positive Sampling. Motivated by the aforementioned drawbacks, we propose a novel dynamic *positive* sampling strategy and combine it with static negative sampling for loss function L_4 defined in Equation 8. The advantage of our approach is two-fold: (1) The dynamic *positive* sampling strategy could drive the entire training process and alleviate the effect of vanishing gradients; (2) It overcomes the performance issue for dynamic negative sampling strategy since both static sampling and dynamic positive sampling can be processed efficiently.

We are now ready to modify loss function L_4 to enable dynamic positive sampling:

$$L_4 = -\sum_{\substack{m=n \\ or \\ u_m,u_n\sim\tilde{p}(m,n)}} \log\sigma(\tilde{\mathbf{x}}_m^{\mathrm{T}} \cdot \tilde{\mathbf{z}}_n) - \sum_{n'=1} \mathbb{E}_{v_{n'}\sim P'_n(v)} \log\sigma(-\tilde{\mathbf{x}}_m{}^{\mathrm{T}} \cdot \tilde{\mathbf{z}}_{n'}), \tag{9}$$

where $\tilde{p}(m,n)$ stands for the dynamic positive sampling distribution and $P'_n(v)$ remains to be the static sampling distribution. Through dynamic positive sampling, we can draw samples that are potentially closed in the latent space and achieve better training performance.

In the cause of reducing computation complexity, we employ HNSW [27] for constructing indexes to achieve efficient dynamic positive sampling. Specifically, for $\tilde{\mathbf{x}}$ of each node, we search its k-nearest neighbors in the latent space through HNSW indexes. The k-nearest neighbors are returned as positive cases and we push their $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$ closer in the embedding space to maintain these positive correlations. The same goes for $\tilde{\mathbf{z}}$. HNSW has been proved to be efficient with lower dimensionality (usually no larger than 32). Henceforth, we enforce the aforementioned transforming kernels to reduce the dimension of latent representations into a small value (e.g., 16, 32).

We combine all optimization functions together as our final optimization function that jointly optimizes the embedding vectors:

$$L = L_1 + L_2 + L_3 + L_4. \tag{10}$$

Algorithm 1 Learning algorithm for BiANE

Input: $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E}, \mathbf{X}_{\mathcal{U}}, \mathbf{X}_{\mathcal{V}})$, embedding dimension d, learning rate η, positive sample number k.
Output: node embedding $\mathbf{H} \in \mathbb{R}^d$.

1: Construct two partitions $\mathcal{U}$ and $\mathcal{V}$
2: Intialize all model parameters θ
3: Obtain the structure feature via Eq. 3
4: **while** not converged **do**
5: Feed $\mathbf{x}$ and $\mathbf{z}$ into autoencoders to obtain $\mathbf{x}'$, $\mathbf{z}'$, $\hat{\mathbf{x}}$, and $\hat{\mathbf{z}}$
6: Update autoencoders parameters via Eq. 4
7: Update encoders parameters via Eq. 5
8: Transform to $\mathbf{x}'$ and $\mathbf{z}'$ to $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$ via Eq. 6
9: Build up HNSW indexes based on $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$
10: Perform k-NN positive sampling for each $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$
11: Update encoders and kernels parameters via Eq. 9
12: Concatenate $\mathbf{x}'$ and $\mathbf{z}'$ to obtain embedding $\mathbf{h}$
13: Update encoders parameters via Eq. 2
14: **end while**
15: **return** node embedding $\mathbf{H}$

We employ static negative sampling for L_1, L_3 and L_4. The dynamic positive sampling strategy is devised for L_4, which is used to drive the training process efficiently.

The training process of BiANE is summarized in Algorithm 1. Lines 5-6 compress the attribute information and the structure information through autoencoders. Line 7 models the first-order proximity within each partition. Lines 8-11 perform dynamic positive sampling in the latent space and model the correlation between the attribute information and the structure information. Lines 12-13 model the inter-partition proximity.

3.7 Complexity Analysis

Assuming that there exists n nodes within the bipartite attributed network and each node is associated with m edges on average, thus compact feature learning (Eq. 4) costs $O(n)$ time and inter-partition modeling (Eq. 2) takes $O(nm)$ complexity. After generating synthetic networks within each partition, we will have nm^2 intra-partition links, which means the time complexity for joint modeling (Eq. 5) is $O(nm^2)$. For correlation training, both the construction and search procedures for HNSW take $O(n \log(n))$ time. Therefore, the overall time complexity of BiANE is $O(n + nm + nm^2 + 2n \log(n))$, which can be simplified as $O(n \log(n))$ since $m \ll n$ for most networks. As a result, the efficiency and scalability of our solution have been validated.

4 EXPERIMENTS

In this section, we first describe our experimental datasets and the competitor baselines. Then, we conduct experiments on link prediction and node classification to demonstrate the efficacy and efficiency of our proposed BiANE model.

4.1 Dataset Descriptions

We evaluate the performance of our proposed model on three real-world datasets. The statistics of the datasets are summarized in Table 2 (Sparsity is computed as $1 - \frac{\#link}{\#user \times \#item}$).

1) MovieLens Dataset[1]. MovieLens contains over 1,000,000 ratings for public movies from anonymous users. Each user is associated with a profile with three categorical features while each movie is classified into several categories, which we deemed as attributes for each node. All ratings from the dataset are made on a 5-star scale (whole-star ratings only). Following the experimental setting of [25, 40, 48], we only regard 5-star ratings as links (or positive feedback), and all other ratings as non-links.
2) AMiner Dataset[2]. AMiner [35] is a free online academic dataset, which contains information of more than 2,000,000 papers and 1,700,000 researcher profiles. We classify the venues of papers to eight categories as metapath2vec [5]. We sample papers with abstract information from the dataset as well as relevant authors, and we treat the abstract and authors' research interests as the attribute information for papers and authors respectively. Following the setting of [44], we transform the attribute information into continuous vectors by Doc2Vec [21].
3) Alibaba Dataset. Alibaba dataset is collected from user activity logs from Taobao website, a world-leading e-commerce platform. The dataset contains over 1,000,000 users and 21,000 items and each user or item is associated with a profile, which includes both categorical and numerical features. We randomly sample a subgraph from the dataset to conduct our experiments.

Table 2: Dataset Statistics

Dataset	#user	#item	#link	#user-attr	#item-attr	sparsity(%)
MovieLens	6,000	3,069	225,344	3	1	0.9878
AMiner	80,461	66,107	168,525	1	1	0.9999
Alibaba	38,140	7,913	59,237	18	28	0.9998

4.2 Baseline Methods

To show the superior performance of the proposed models, we compare BiANE with state-of-the-art network embedding methods. The details of these methods are listed as follows.

- **DeepWalk [29]**: DeepWalk employs random walks to get node sequences and treat them as word corpus. A Skip-gram model [28] is then applied on them to obtain node embeddings.
- **node2vec [13]**: This work defines a flexible notion to perform biased random walks to obtain node sequences and utilizes a Skip-gram model [28] to learn node embeddings.
- **SDNE [37]**: SDNE adopts autoencoders to compress the information of the node's local neighborhood and jointly models the first-order proximity and the second-order proximity to encode the network structure.
- **metapath2vec++ [5]**: metapath2vec++ treats node sequences on predefined metapaths as word corpus and then utilizes a Skip-gram model [28] and a heterogeneous negative sampling strategy to learn node embeddings.
- **BiNE [10]**: BiNE is the state-of-the-art model for representation learning on bipartite networks. It models the proximity across two partitions and the node proximity within the same partition. Since it doesn't support node attributes, only the structure proximity can be captured.

[1] https://grouplens.org/datasets/movielens/
[2] https://aminer.org/aminernetwork

- **NGCF [43]**: NGCF leverages high-order connectivities in the user-item interaction graph and adopts an embedding propagation mechanism to harvest the collaborative signal from recommender systems.
- **ANRL [50]**: ANRL develops a neighborhood enhancement autoencoder to extract the attribute information and designs an attribute-aware skip-gram model to preserve the direct and indirect neighborhood proximity for embedding networks.
- **AANE [15]**: AANE is fabricated in a distributed manner for attributed network embedding at scale. It jointly performs attribute affinity decomposition and penalizes the embedding difference between connected nodes.
- **FeatWalk [16]**: FeatWalk is a scalable framework that deals with the heterogeneous feature embedding problem. It incorporates multiple instance feature matrices and network structures to generate feature-based node sequences and use a Skip-gram model [28] for node embeddings.
- **STAR-GCN [47]**: STAR-GCN is a recently proposed GCN extended model for recommender systems that supports node attributes. It adopts a stacked and reconstructed GCN architecture and employs a mask technique to learn node embeddings.

4.3 Implementation Details

For all baselines, we used the default implementation released by the original authors and the parameters are tuned for optimal performance. Noticing that ANRL, AANE, and FeatWalk targets for embedding homogeneous attributed networks and have one encoder for node attributes only, we concatenate attributes from two partitions by padding them with 0 to a fixed length. To compare fairly, we set the embedding dimension as 128 for all methods.

Table 3: #neurons in each layer for attribute autoencoders

Datasets	MovieLens	AMiner	Alibaba
Partition $\mathcal{U}$	30-48-64-48-30	128-64-128	94-64-94
Partition $\mathcal{V}$	18-42-64-42-18	128-64-128	73-64-73

For BiANE, we randomly initialize model parameters with a standard Xavier normal distribution [12] and optimizing the model with Adam [19]. To obtain high-order structure features for each partition, we adopt metapath2vec++ [5] via two meta-path schemes ($\mathcal{U}-\mathcal{V}-\mathcal{U}$ or $\mathcal{V}-\mathcal{U}-\mathcal{V}$) to obtain a 128-dimension vectors as $\mathbf{z}$ for each node within its partition. Thus, we set the neurons for structure autoencoder for both partitions in all datasets as 128-64-128. The number of attributes varies for different datasets and we list the parameters for attribute autoencoders in Table 3. The transforming kernels for all datasets are set as 64-16 to reduce the representation dimension to 16 for HNSW k-NN search. The number of HNSW [3] dynamic sampling cases k is set as 5 for all datasets. The dimension of the final embedding vector $\mathbf{h}$ is set to be 128, the same as other baseline methods for a fair comparison.

4.4 Efficacy Study

4.4.1 Link Prediction. Link prediction is a typical task for network representation learning. The goal is to predict whether there exists a link between two given nodes. For each dataset, We randomly hold out 70% links for training, 10% as the validation set, and the remaining 20% links are treated as the test set. Since the validation/test set contains only positive cases (links), we randomly sample the same number of negative cases (non-links) for validation and test. After training, we use logistic regression to predict the probability of a link. Following previous works [10, 11, 45], we employ two metrics, the area under the ROC curve (AUC-ROC) and the area under the Precision-Recall curve (AUC-PR) to evaluate the performance.

Table 4: Link Prediction Results

Model	MovieLens		AMiner		Alibaba	
	AUC (ROC)	AUC (PR)	AUC (ROC)	AUC (PR)	AUC (ROC)	AUC (PR)
DeepWalk	0.6583	0.6229	0.7730	0.8378	0.8074	0.8353
node2vec	0.6597	0.6296	0.8169	0.8649	0.8605	0.8846
SDNE	0.7454	0.7393	0.5638	0.5646	0.5863	0.6267
metapath2vec++	0.7243	0.6736	0.6935	0.7480	0.8188	0.8346
BiNE	0.7616	0.7297	0.5997	0.5812	0.6886	0.6411
NGCF	0.7547	0.7117	0.7692	0.8290	0.8574	0.8856
ANRL	0.5554	0.5449	0.8350	0.8251	0.6639	0.6429
AANE	0.7010	0.6670	0.5943	0.5924	0.7142	0.6852
FeatWalk	0.7117	0.7007	0.7589	0.8086	0.7948	0.8180
STAR-GCN	0.7621	0.7405	0.6455	0.6587	0.5924	0.5721
BiANE	**0.7711**	**0.7409**	**0.8972**	**0.9054**	**0.8903**	**0.8997**

The link prediction results are shown in Table 4. The best result in each case is highlighted in bold. We can observe that the proposed BiANE outperforms all compared baselines, and some key observations are presented as follows:

- For homogeneous network embedding methods, DeepWalk and node2vec perform well on Alibaba and AMiner datasets, while they achieve relatively inferior results on MovieLens. This is because nodes in MovieLens are more densely connected and random walk based approaches introduce noises when the walk reaches several hops away from the starting node. In contrast, SDNE achieves better results on MovieLens as it focuses more on local neighborhoods by explicitly employing the first-order and second-order proximity. Nevertheless, SDNE under-performs in AMiner and Alibaba datasets as it can only extract limited information from the local neighborhood in sparse networks.
- For heterogeneous embedding methods, metapath2vec++ and BiNE do not achieve competitive results compared to homogeneous embedding methods, except for the MovieLens dataset. The reason is that for dense bipartite networks, the node type information is more useful than the sparse networks to reduce noise introduced by random walks. NGCF obtains competitive results in three datasets. This is mainly attributed to its successful modeling of high-order connectivities facilitates the preservation of collaborative signal, which has exactly validated the rationality of our modeling of the structure proximity within each partition.
- For attributed network embedding methods, ANRL, AANE, and FeatWalk obtain relatively stable results. Since they do not consider to differentiate the attribute information according to partitions, they introduce more noise into embeddings. Thus, they demonstrate inferior results. STAR-GCN achieves high AUC value in MovieLens while obtaining unsatisfactory results in

[3] We use the implementation from nmslib: https://github.com/nmslib/nmslib

Table 5: Node Classification Results

Model	AMiner				Alibaba			
	60%		80%		60%		80%	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
DeepWalk	0.4427	0.2689	0.4440	0.2736	0.3800	0.1194	0.3866	0.1026
node2vec	0.4587	0.2907	0.4583	0.2898	0.3661	0.0992	0.3879	0.1001
SDNE	0.2833	0.1141	0.2842	0.1132	0.3800	0.0919	0.3765	0.0907
metapath2vec++	0.3926	0.2183	0.3925	0.2199	0.3809	0.1122	0.3860	0.1030
BiNE	0.2648	0.1074	0.2648	0.1067	0.4011	0.0828	0.3999	0.0828
NGCF	0.3417	0.0968	0.3408	0.1094	0.4005	0.0818	0.3986	0.0850
ANRL	0.7772	0.6777	0.7778	0.6779	0.4015	0.0818	0.3992	0.0815
AANE	0.7574	0.6651	0.7550	0.6616	0.3986	0.0912	0.3967	0.0913
FeatWalk	0.3779	0.1977	0.3819	0.2009	0.3759	0.1581	0.3910	0.1554
STAR-GCN	0.2951	0.1278	0.2938	0.1276	0.4008	0.0818	0.3980	0.0814
BiANE	**0.8000**	**0.7137**	**0.7976**	**0.7115**	**0.4078**	**0.1866**	**0.4245**	**0.1795**

Table 6: Node Classification Results for Ablation Study

Model	AMiner				Alibaba			
	60%		80%		60%		80%	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
BiANE-ATTR	0.7931	0.7089	0.7925	0.7069	0.4062	0.1423	0.4024	0.1327
BiANE-STRUC	0.3818	0.2047	0.3841	0.2077	0.3958	0.0888	0.3961	0.0851
BiANE-INTER	0.7961	0.7083	0.7924	0.7059	0.3977	0.1691	0.4144	0.1673
BiANE-CONCAT	0.7973	0.7063	0.7949	0.7032	0.4065	0.1798	0.4125	0.1646
BiANE-LAYER	0.7967	0.7093	0.7947	0.7051	0.3986	0.1754	0.4087	0.1701
BiANE-IS	0.7970	0.7118	0.7939	0.7075	0.4015	0.1786	0.4201	0.1755
BiANE-ISL	0.7985	0.7079	0.7966	0.7057	**0.4087**	0.1849	0.4131	0.1726
BiANE	**0.8000**	**0.7137**	**0.7976**	**0.7115**	0.4078	**0.1866**	**0.4245**	**0.1795**

Table 7: Link Prediction Results for Ablation Study

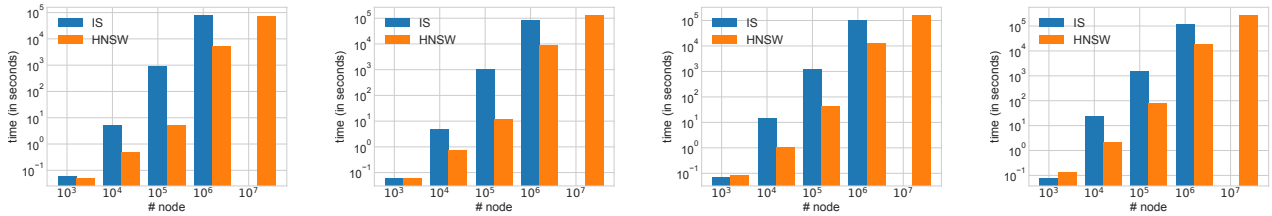
Model	MovieLens		AMiner		Alibaba	
	AUC (ROC)	AUC (PR)	AUC (ROC)	AUC (PR)	AUC (ROC)	AUC (PR)
BiANE-ATTR	0.6007	0.5892	0.8761	0.8692	0.7433	0.7216
BiANE-STRUC	0.7593	0.7270	0.7305	0.7817	0.8806	0.8911
BiANE-INTER	0.7423	0.7016	0.8915	0.9022	0.8881	0.8965
BiANE-CONCAT	0.7658	0.7348	0.8935	0.9033	0.8873	0.8946
BiANE-LAYER	0.7674	0.7378	0.8935	0.9032	0.8861	0.8923
BiANE-IS	0.7690	0.7393	0.8942	0.9043	**0.8903**	**0.9000**
BiANE-ISL	0.7690	0.7385	0.8957	**0.9057**	0.8902	0.8996
BiANE	**0.7711**	**0.7409**	**0.8972**	0.9054	**0.8903**	0.8997

AMiner and Alibaba. The main reason is that its key insights is to aggregate information from local neighbors and is bound to suffer from sparsity issues, which is in line with SDNE.

- BiANE achieves the best result. Since our model can capture the high-order structure proximity and the first-order proximity within intra-partition, our model is more robust to network sparsity. Besides, our model leverages the auxiliary information from node attributes, which can bring further improvements to embedding results. Hence, preserving the intra-partition proximity for each partition can contribute more constructive information to enhance the performance of the proposed model.

4.4.2 Node Classification. Node Classification is another widely used method for evaluating the performance of network embedding. To evaluate comprehensively, we randomly select {60%, 80%} nodes to train a logistic regression classifier and then test the classifier performance on the rest of the nodes. To measure the performance of the classification task, we use Micro-F1 and Macro-F1 for evaluation. For the methods that require attribute encoding (ANRL, AANE, FeatWalk, STAR-GCN, BiANE), we do not involve ground-truth labels of the classification tasks. We present the classification results of *item-purchase-level* for Alibaba dataset, and *paper-venue* class for AMiner dataset. Since MovieLens dataset only contains a single label for each node, we do not conduct experiments on the MovieLens dataset due to insufficient attribute information.

The results are shown in Table 5 and the best results are marked in bold. STAR-GCN and SDNE again perform badly on these datasets for they fail to tackle the sparsity problem, even though STAR-GCN is able to aggregate the attribute information. For other methods



(a) dimension-16 (b) dimension-64 (c) dimension-128 (d) dimension-300

Figure 4: Efficiency of dynamic sampling strategy w.r.t network volume. We do not report the running time for Importance Sampling at the scale of 10^7 nodes for it could not complete a single round of sampling within a week.

that can handle associated network attributes, we can observe that they present competitive results for using the attribute information. FeatWalk's performance on AMiner dataset is an exception. The reason is that the attribute information for AMiner dataset is numerical (note that we adopt Doc2Vec [21] to obtain features for AMiner dataset), FeatWalk can not generate good feature-based node sequences. BiANE again achieves the best results. This is mainly because BiANE successfully leverages the attribute information and differentiates them according to different partitions. In addition, the latent correlation training within each partition can enhance the integration of the attribute information and the structure information, which also accounts for the superior performance of our proposed model.

4.5 Ablation Study

We conduct an ablation study to evaluate each part of our model. We configure BiANE to the following settings:

- **BiANE-ATTR**: We utilize the attribute information from the network only and do not explicitly consider the structure information in the intra-partition proximity modeling,
- **BiANE-STRUC**: We model the intra-partition proximity without incorporating the attribute information.
- **BiANE-INTER**: We only perform the compact feature learning and the inter-partition proximity modeling of the network.
- **BiANE-CONCAT**: We do not perform latent correlation training and combine the attribute encoding with structure encoding through direct concatenation.
- **BiANE-LAYER**: We do not perform latent correlation training and integrate the attribute encoding and structure encoding by feeding them through neural layers.
- **BiANE-IS**: We directly train the attribute-structure correlation on the attribute encoding space and the structure encoding space. The sampling strategy is importance sampling with a sampling probability distribution $\frac{\exp(\tilde{p}(m,n))}{\sum_{n'}\exp(\tilde{p}(m,n'))}$, and the number of positive sampling cases k is also set as 5.
- **BiANE-ISL**: We perform latent correlation training on the latent space from the output of transforming kernels. The sampling strategy is the same as **BiANE-IS**.

We conduct experiments on link prediction and node classification, and the results are presented in Table 7 and 6. The best results are highlighted in bold and second best results are underlined. We can observe that incorporating both the attribute information and the structure information can enhance the representation learning results. BiANE-INTER shows inferior performance on both tasks compared to the ones with the intra-partition proximity modeling. BiAN-CONCAT, BiANE-LAYER combine attribute encoding and structure encoding during the intra-partition proximity modeling, they show better performance compared to the former configured models. BiANE-IS, BiANE-ISL achieve competitive results among all configured models, proving the rationality of our attribute-structure correlation modeling and our intuition of dynamic sampling strategy. Specifically, BiANE-ISL is generally better than BiANE-IS, confirming that the attribute-structure correlation training should be better performed on a separate latent space, so as to circumvent the potential disturbance and instability problems introduced by modeling correlation directly on the encoding space.

Generally speaking, BiANE achieves the best results on most experiments among all configured models, which demonstrates that the HNSW-based dynamic positive sampling strategy can also reach better or similar improvements on network embeddings as the importance sampling strategy. Compared to BiANE-ISL, our proposed dynamic sampling strategy is more robust and noise-intolerant. Thus, the superiority of BiANE has been proved.

4.6 Efficiency Study

The dynamic sampling strategy is heavily affected by the size of the network, so we synthesize networks of different scales to test the efficiency of the sampling strategy. We generate random graphs with nodes number ranging from 10^3 to 10^7 and perform our HNSW-based dynamic sampling strategy to evaluate the efficiency. We also present the performance of the importance sampling strategy with a sampling probability $\frac{\exp(\tilde{p}(m,n))}{\sum_{n'}\exp(\tilde{p}(m,n'))}$ as the baseline. The embedding dimension is set as [16, 64, 128, 300] for experiments and the number of sampling cases k is set as 10. All experiments are conducted on a single Linux server with 40 Intel(R) Xeon(R) CPU (Gold 6138 CPU @ 2.00GHz) and 512G memory. To fairly evaluate the efficiency, either strategy is tested on a single core with a single thread during the running process. We report the running time (both IO and computation time) as the metric for evaluation and results are shown in Figure 4. We can observe that HNSW-based sampling is more efficient than importance sampling. Importance sampling even could not complete a single round of sampling with a network of 10^7 nodes within a week. Specifically, the smaller the dimension is, the more efficient the strategy is, which proves that the efficiency of our proposed approach.

5 CONCLUSION

In this work, we present BiANE, a novel model that targets to tackle the challenges of bipartite attributed network embedding. BiANE augments intra-partition links for bipartite attributed networks and employs autoencoders to aggregate the attribute and structure information within each partition and perform *latent correlation training* to preserve their highly non-linear relationship in a new latent space. Moreover, we adopt an efficient *dynamic positive sampling strategy* to ameliorate the training process with competitive complexity. Comprehensive experiments are conducted on several real-world networks and the results demonstrate that BiANE can achieve significant gains against state-of-the-art methods.

ACKNOWLEDGMENTS

This research is supported by Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, the National Research Foundation Singapore, and AI Singapore under its Research Programme (AISG-RP-2018-001). Ju Fan is supported by the NSFC (61632016, U1711261) and the Research Funds of Renmin University of China (18XNLG18).

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*.
[2] Chiara Carusi and Giuseppe Bianchi. 2019. A look at interdisciplinarity using bipartite scholar/journal networks. *Scientometrics* (11 2019). https://doi.org/10.1007/s11192-019-03309-3
[3] Long Chen, Fajie Yuan, Joemon M. Jose, and Weinan Zhang. 2018. Improving Negative Sampling for Word Representation using Self-embedded Features. In *WSDM*. ACM, 99–107.
[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *DLRS@RecSys*.
[5] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *KDD*.
[6] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NIPS*.
[7] Tao-Yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*.
[8] Hongchang Gao and Heng Huang. 2018. Deep Attributed Network Embedding. In *IJCAI*.
[9] Hongchang Gao and Heng Huang. 2018. Self-Paced Network Embedding. In *KDD*.
[10] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding. In *SIGIR*.
[11] Ming Gao, Xiangnan He, Leihui Chen, and Aoying Zhou. 2019. Learning Vertex Representations for Bipartite Networks. *CoRR* abs/1901.09676 (2019).
[12] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS (JMLR Proceedings)*, Vol. 9.
[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *KDD*.
[14] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *IJCAI*.
[15] Xiao Huang, Jundong Li, and Xia Hu. 2017. Accelerated Attributed Network Embedding. In *SDM*.
[16] Xiao Huang, Qingquan Song, Fan Yang, and Xia Hu. 2019. Large-Scale Heterogeneous Feature Embedding. In *AAAI*.
[17] Xuqing Huang, Irena Vodenska, Shlomo Havlin, and Harry Eugene Stanley. 2013. Cascading Failures in Bi-partite Graphs: Model for Systemic Risk Propagation. In *Scientific reports*.
[18] Meng Jiang, Peng Cui, Nicholas Jing Yuan, Xing Xie, and Shiqiang Yang. 2016. Little Is Much: Bridging Cross-Platform Behaviors through Overlapped Crowds. In *AAAI*.
[19] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.
[20] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR (Poster)*.
[21] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML*. 1188–1196.
[22] Yuchen Li, Ju Fan, George V. Ovchinnikov, and Panagiotis Karras. 2019. Maximizing Multifaceted Network Influence. In *ICDE*. IEEE, 446–457.
[23] Yuchen Li, Ju Fan, Yanhao Wang, and Kian-Lee Tan. 2018. Influence Maximization on Social Graphs: A Survey. *IEEE Trans. Knowl. Data Eng.* 30, 10 (2018), 1852–1872.
[24] Lizi Liao, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2018. Attributed Social Network Embedding. *IEEE Trans. Knowl. Data Eng.* 30, 12 (2018).
[25] Qiuxia Lu, Tianqi Chen, Weinan Zhang, Diyi Yang, and Yong Yu. 2012. Serendipitous Personalized Ranking for Top-N Recommendation. In *Web Intelligence*.
[26] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *WSDM*.
[27] Yury A. Malkov and D. A. Yashunin. 2016. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *CoRR* abs/1603.09320 (2016).
[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*.
[29] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *KDD*.
[30] Paola Pesantez-Cabrera and Ananth Kalyanaraman. 2016. Detecting Communities in Biological Bipartite Networks. In *BCB*. ACM, 98–107.
[31] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.
[32] Ruslan Salakhutdinov and Geoffrey E. Hinton. 2009. Semantic hashing. *Int. J. Approx. Reasoning* (2009).
[33] Gábor Takács and Domonkos Tikk. 2012. Alternating least squares for personalized ranking. In *RecSys*.
[34] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*.
[35] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: extraction and mining of academic social networks. In *KDD*.
[36] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *CoRR* abs/1706.02263 (2017).
[37] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *KDD*.
[38] Hongyang Wang, Qingfei Meng, Ju Fan, Yuchen Li, Laizhong Cui, Xiaoman Zhao, Chong Peng, Gong Chen, and Xiaoyong Du. 2020. Social Influence Does Matter: User Action Prediction for In-Feed Advertising. In *AAAI*.
[39] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. GraphGAN: Graph Representation Learning With Generative Adversarial Nets. In *AAAI*.
[40] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *SIGIR*.
[41] Keqiang Wang, Hongwei Peng, Yuanyuan Jin, Chaofeng Sha, and Xiaoling Wang. 2016. Local Weighted Matrix Factorization for Top-n Recommendation with Implicit Feedback. *Data Science and Engineering* 1, 4 (2016), 252–264.
[42] PeiFeng Wang, Shuangyin Li, and Rong Pan. 2018. Incorporating GAN for Negative Sampling in Knowledge Representation Learning. In *AAAI*.
[43] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. ACM, 165–174.
[44] Yueyang Wang, Ziheng Duan, Binbing Liao, Fei Wu, and Yueting Zhuang. 2019. Heterogeneous Attributed Network Embedding with Graph Convolutional Networks. In *AAAI*.
[45] Shuang Xia, Bing Tian Dai, Ee-Peng Lim, Yong Zhang, and Chunxiao Xing. 2012. Link Prediction for Bipartite Social Networks: The Role of Structural Holes. In *ASONAM*.
[46] Lu Yu, Chuxu Zhang, Shichao Pei, Guolei Sun, and Xiangliang Zhang. 2018. WalkRanker: A Unified Pairwise Ranking Model With Multiple Relations for Item Recommendation. In *AAAI*.
[47] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. 2019. STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems. In *IJCAI*. ijcai.org, 4264–4270.
[48] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *SIGIR*.
[49] Yongqi Zhang, Quanming Yao, Yingxia Shao, and Lei Chen. 2019. NSCaching: Simple and Efficient Negative Sampling for Knowledge Graph Embedding. In *ICDE*. IEEE, 614–625.
[50] Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang. 2018. ANRL: Attributed Network Representation Learning via Deep Neural Networks. In *IJCAI*.
[51] Vincent W. Zheng, Mo Sha, Yuchen Li, Hongxia Yang, Yuan Fang, Zhenjie Zhang, Kian-Lee Tan, and Kevin Chen-Chuan Chang. 2018. Heterogeneous Embedding Propagation for Large-Scale E-Commerce User Alignment. In *ICDM*.