TriNE: Network Representation Learning for Tripartite Heterogeneous Networks

Zhabiz Gharibshah and Xingquan Zhu

Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University
Boca Raton, FL 33431, USA
{zgharibshah2017, xzhu3}@fau.edu

Abstract—In this paper, we study network representation learning for tripartite heterogeneous networks which learns node representation features for networks with three types of node entities. We argue that tripartite networks are common in real-world applications, and the essential challenge of the representation learning is the heterogeneous relations between various node types and links in the network. To tackle the challenge, we develop a tripartite heterogeneous network embedding called TriNE. The method considers unique user-item-tag tripartite relationships, to build an objective function to model explicit relationships between nodes (observed links), and also capture implicit relationships between tripartite nodes (unobserved links across tripartite node sets). The method organizes metapath-guided random walks to create heterogeneous neighborhood for all node types in the network. This information is then utilized to train a heterogeneous skip-gram model based on a joint optimization. Experiments on real-world tripartite networks validate the performance of TriNE for the online user response prediction using embedding node features.

Index Terms—machine learning, network representation learning, tripartite heterogeneous networks

I. Introduction

Many real-world applications have tripartite relationships between different sets of entities. For example, e-commence systems usually involve user-product-vendor connections where users purchase products (user-product relations) and products are made by vectors (product-vendor relations). Similarly, health domains often have patient-disease-drug bonds, where patients are diagnosed with different diseases (patient-disease relations), and diseases can be treated by drugs (disease-drug relations).

In a tripartite network, connections between entities are unique and have very interesting (and useful) meanings. For example, in a user-product-vendor tripartite network, the bipartite user-product and product-vendor relations are directly observable, whereas the user-vendor bipartite relationship is often unobservable which commonly refers to the loyalty of users to the brand [1]. On the other hand, in a patient-disease-drug tripartite network, the bipartite patient-disease and disease-drug relations are easy to collect, whereas the patient-drug bipartite relationship is often unknown and is commonly tied to adverse drug relations [2]. Therefore, an accurate link prediction in a tripartite network setting is important to understand how users respond to the third party.

Indeed, the unique three-party (*i.e.*, tripartite) entity relationships provide useful information which is hard to be characterized by traditional heterogeneous information networks. This also provides challenges for existing network embedding methods to be directly applied to the tripartite networks

- **Imbalanced Nodes:** tripartite networks have multiple node types, and often have severely imbalanced nodes where one type of nodes are often much more than other types (*e.g.* number of users are far more than the number of vendors). As a result, a traditional global random walk based network representation learning, such as node2vec [3], cannot learn effective features for tripartite network nodes.
- **Imbalanced network links:** In many tripartite networks, edges between two types of nodes, such as user-clicks on advertisements [4] or user adverse drug reaction on particular drugs [2] are rare events, which are making it difficult to learn effective features for the link prediction task.

While network embedding approaches, including bipartite network embeddings, have been commonly studied recently [5], no work currently exists to learn node representation for tripartite networks explicitly. Particularly in handling the above two challenges for tripartite network embedding learning.

Motivated by the above observations, in this paper, we propose TriNE, a tripartite network embedding learning algorithm to learn features to represent nodes for tripartite heterogeneous networks. TriNE considers coupled bipartite correlations to build an objective function to model explicit relationships between bipartite nodes (observed links between nodes), and also capture implicit relationships between tripartite nodes (unobserved links across tripartite node sets). The optimization allows the metapath-guided random walks carry out on the tripartite network to handle imbalanced node numbers for an effective link prediction. Experiments on real-world tripartite networks validate the performance of the proposed method.

II. Related work

Network representation learning has gained a lot of attentions recently [5]. The classic dimension reduction methods like LLE [6], ISOMAP [7], Laplacian eigenmaps [8] and

their extensions [9], [10] adopted various versions of linear and non-linear matrix factorization based techniques. Although the performance of these methods are shown successful for relatively small networks, but applying matrix decomposition is not scalable for large networks.

The network embedding has shown the decent performance in many data mining and recommendation systems. The initial studies followed the word2vec idea [11] to extend the skip-gram models for homogeneous networks like LINE [12], node2vec [3]. These random walk based methods generate a corpus of random walk samples which are fed into skip-gram models to learn node embedding vectors.

Recently, several studies have been proposed for heterogeneous network embedding. Authors in [13] proposed a method to encode inter- and intra-network edges in heterogeneous networks. In [14], authors extended DeepWalk [15] method by introducing metapath-guided random walks and a heterogeneous skip-gram model. SHNE method which was proposed in [16] investigated to integrate semantic information along with structural relations between nodes into the network embedding procedure.

There are many studies developed based on the architecture of graph neural networks. In [17], authors proposed a metapath-based embedding method mainly designed using a graph neural network(GNN). The representation of the knowledge graph as the type of heterogeneous networks have received a lot of interest recently. Knowledge graphs (KG) are semantic heterogeneous networks including a collection of entities with attributes that are inter-connected together through edges. Here relations corresponding to edges may have different types and functionalities. They are usually described through a triplet like (Head, Relation, Tail) that a relation connects head and tail entities. This structure of data has been studied for different applications like the link prediction. Recently, a study [18] elaborated a GNN-based method for node embedding in knowledge graphs in which weights of link between nodes like user and item in the network are not available. They suggested a supervised learning method to model a personalized scoring function to determine weights that followed a relational heterogeneity principle in the knowledge graph. In order to deal with data sparsity problem, they designed a leave-one-out loss function combined with a label smoothness regularization to predict the weight values of links. They then were used to calculate node embeddings through a local neighborhood aggregation. In many knowledge graph based recommendation systems, the interactions between two entities of user and item are usually modelled by applying aggregation mechanisms like average pooling and attention units over their user-defined neighborhood based on calculated node embedding vectors. Then, a sigmoid function with inner-product kernel is used to represent the probability of links [12], [19]. The authors in [20] proposed a neighborhood interaction model to integrate high order neighbor-neighbor interactions for training in a graph neural network. They improved the performance of predicting user click-through rate values as the link between user and

item entities using a bi-attention network in their design.

In [21] authors suggested an end-to-end learning method called RippleNet to unify the user preference propagation within a knowledge graph embedding to calculate the distribution of user interests regarding an item. In this paper, we intend to propose a network representation method to embed a tripartite heterogeneous network into a low-dimensional vector space.

III. Preliminaries & Problem Definition

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ denotes a heterogeneous network (graph) which includes a vertex set ($\mathcal{V}$), an edge set $\mathcal{E}$ with different types. They are characterized with two type mapping functions as $\phi : \mathcal{V} \rightarrow O$ and $\psi : \mathcal{E} \rightarrow R$ where O is the set of node types and R is the set of edge types in the network. Each vertex(entity) $v \in \mathcal{V}$ in the graph is associated with vertex type using $\phi(v) \in O$. Similarly, each edge $e \in \mathcal{E}$ belongs to an edge type specified by $\psi(e) \in R$. A tripartite heterogeneous graph is the specific type of the heterogeneous graph where $|O| = 3$ and $|R| = 3$. The node set $\mathcal{V}$ consists of three node types $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{V}_3$, which in turn encompasses three types of entities in the tripartite network, such as user nodes ($\mathcal{V}_1$), page categories ($\mathcal{V}_2$), and item category nodes ($\mathcal{V}_3$) in a tripartite user interaction network. Due to the tripartite network nature, we assume network edges always connect nodes between two different parties, i.e. $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{E}_3$ where $E_1 = (\mathcal{V}_1 \times \mathcal{V}_2)$, $\mathcal{E}_2 = (\mathcal{V}_2 \times \mathcal{V}_3)$ and there is $\mathcal{E}_3 = (\mathcal{V}_1 \times \mathcal{V}_3)$. In the case of recommendation systems and online advertising, the portion of links between $\mathcal{V}_1$ and $\mathcal{V}_3$ are missing. Therefore, the connectivity between users and items(Ads) is generally evaluated as the target [19], [22], [23].

In the heterogeneous network, dealing with different types of nodes brings in a semantic relationship between nodes and edges. This complexity is generally described with a meta-data modeling through the network schema and metapaths.
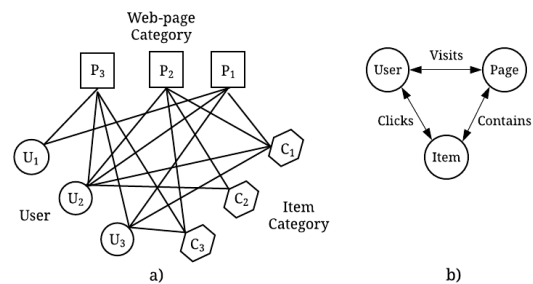

Fig. 1. a) An example of heterogeneous network including three types of nodes and inter-connecting edges b) Network schema

Formally, metapath is defined as a semantic path in the heterogeneous graph. Following the network schema in this case, the path is shown as a sequence of different relations between node types as: $A_0 \xrightarrow{R_0} A_1 \xrightarrow{R_1} A_2 \dots \xrightarrow{R_n} A_{n+1}$ where $R = R_0 R_1 R_2 \dots R_n$ describes a composite relation between node type A_0 and A_{n+1}.

Figure 1 shows an example of a tripartite network along with the corresponding network schema. Different metapaths

represent different semantic information. For example, meta-path "UPC" indicates a user visiting a page category which contains an item's category. Moreover, "UPCPU" represents two users who visit page categories including the same item category type. With regard to metapaths, metapath-guided random walks and metapath-guided neighbors concepts are also introduced [24], [17]. The metapath-guided random walk is considered as a random walk started from any node in the network which follows the specific metapath to randomly select the next node of the walk. The generation of the next node of the random walk is continuously repeated over the metapath scheme until reaching to the user-defined length. Metapath-guided neighbors are similarly defined as the visited nodes when generating random walks followed by a given metapath. In this case, local neighborhood around nodes are defined according to the step travelled from a specific source node. So the neighbors of each node is considered as the combination of the visited nodes till given number of steps are taken. For example, in Figure 1, if we denote i-step neighbors of the node x guided by the metapath ρ as $N_\rho^i(x)$, so $N_\rho^1(U_1) = \{P_3, P_1\}$, $N_\rho^2(U_1) = \{U_2, U_3, C_3, C_1\}$. Then all metapath-guided neighbors of the node U_1 are $N_\rho(U_1) = \{P_3, P_1, U_2, U_3, C_3, C_1\}$. Several studies in literature showed that metapaths and extended versions of metapaths i.e meta-graphs can be seen as a tool to define the proximity and similarity measures [25], [26], [27]. In this case, given a tripartite heterogeneous graph, the goal of problem in this paper is defined as the encoding of entities in the form of embedding vectors by using a proposed embedding approach.

In the following section, we present a heterogeneous network embedding called TriNE for tripartite heterogeneous networks.

A. Tripartite heterogeneous network embedding

In order to learn the representation of nodes, algorithms like DeepWalk [15] and node2vec [3] utilize the skip-gram model followed by random walks to map the concepts of words and context in word2vec [11] to networks. As those methods are designed to handle homogeneous networks, inspired by the recent progress on the network representation learning like BiNE [28] to deal with bipartite networks, we formulate the network representation learning method for tripartite heterogeneous networks through the skip-gram model to account the implicit and explicit relations between nodes.

Figure 2 presents the overall workflow of our proposed method. We consider a tripartite heterogeneous graph as the input. The proposed method includes two major steps regarding capturing explicit and implicit information in the network. Explicit relations are observed edges between three types of node and implicit relations are transitive relations for different type nodes. They are characterized using metapath-guided neighbors gathered from metapath-guided random walks. In the embedding approach both explicit and implicit relations are addressed through a joint optimization. Given tripartite graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, the goal is defined to learn a map function to project each node $v \in \mathcal{V}$ in tripartite network to a low-dimensional numerical vector $e_v \in \mathbb{R}^d$. It is expected that the learned embedding vectors can incorporate informative features to be used in the down-stream task like the predicting links in the network.

B. Implicit relation modeling

Using the skip-gram architecture to model the network, the context of each node v is defined as the set of surrounding nodes $N_c(v)$ connected to it through edges. For the embedding of each node in the homogeneous graph, the probability to see independently each member of $N_c(v)$ in the embedded space can be defined as

$$P(N_c(v) \mid v) = \prod_{v_x \in N_c(v)} P(v_x \mid v; \theta) \qquad (1)$$

Therefore, the objective function is defined as a maximum likelihood problem to maximize the probability to observe the vector neighborhood as the context of each node in its feature representation. According to the three types of nodes in tripartite networks such as $\{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$, three likelihood functions can be defined separately. For the all nodes within set $\mathcal{V}_1$, the first likelihood function is:

$$O_1 = \underset{\theta}{\arg\max} \prod_{u_i \in \mathcal{V}_1} \prod_{u_x \in N_{c(\mathcal{V}_1)}(u_i)} P(u_x \mid u_i; \theta) \qquad (2)$$

where θ includes the embedding vectors of all nodes from the type of $\mathcal{V}_1$. $N_{c(\mathcal{V}_1)}(u_i)$ also denotes all $\mathcal{V}_1$-type context nodes connected to the node u_i via metapath-guided random neighbors in the network. In this case, $P(u_x \mid u_i; \theta)$ is usually approximated as a heterogeneous softmax function using the inner product kernel [3], *i.e.*:

$$P(u_x \mid u_i; \theta) = \frac{e^{\theta_{u_x}^T \theta_{u_i}}}{\sum_{u_k \in \mathcal{V}_1} e^{\theta_{u_k}^T \theta_{u_i}}} \qquad (3)$$

The maximization of the above optimization function is computationally expensive because all nodes are used for the calculation of the denominator in the softmax function. The general approach is to adopt a negative sampling which randomly samples relatively small set of nodes from some pre-defined distributions to approximate the denominator of softmax function. Here we employ a locality sensitive hashing method [28] as follows: Given a node in the network, by defining the bucket of its neighbors based on the topological structure of tripartite network, negative samples are chosen from the remaining nodes different from the neighbors of the node. Therefore, instead of using softmax function to parameterize $P(u_x \mid u_i; \theta)$ in Eq. (2), the following equation is used:

$$P(u_x \mid u_i; \theta) = \prod_{z \in \{u_x\} \cup N_{e(\mathcal{V}_1)}(u_i)} P(z \mid u_i; \theta) \qquad (4)$$

where $N_{e(\mathcal{V}_1)}(u_i)$ shows the negative samples of the node u_i sampled among the $\mathcal{V}_1$ set and $P(z \mid u_i; \theta)$ is also defined as:

$$P(z \mid u_i; \theta) = \begin{cases} \sigma(\theta_{u_i}^T \theta_z) & \text{if z is context of } u_i \\ 1 - \sigma(\theta_{u_i}^T \theta_z) & \text{z} \in N_{e(\mathcal{V}_1)}(u_i) \end{cases} \qquad (5)$$
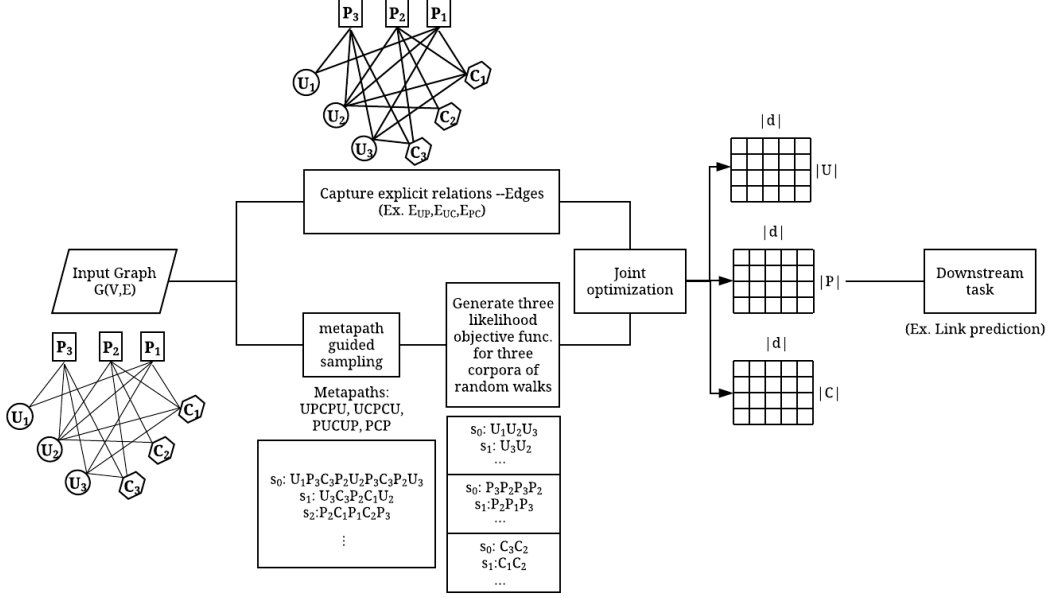
499

Fig. 2. Tripartite network embedding structure. The algorithm includes two major components to capture explicit relations (edges) and implicit relations between nodes. The result is fused using a joint optimization to generate node embedding vectors.

In the similar way, for each node $v \in \mathcal{V}_2$ and $w \in \mathcal{V}_3$, the optimization function can be shown as:

$$O_2 = \underset{\theta}{\arg\max} \prod_{v \in \mathcal{V}_2} \prod_{v_x \in N_{c(\mathcal{V}_2)}(v)} P(v_x \mid v; \theta) \qquad (6)$$

$$O_3 = \underset{\theta}{\arg\max} \prod_{w \in \mathcal{V}_3} \prod_{w_x \in N_{c(\mathcal{V}_3)}(w)} P(w_x \mid w; \theta) \qquad (7)$$

where $P(v_x \mid v; \theta)$ and $P(w_x \mid w; \theta)$ are also parameterized by the function introduced in Eq. (4) and (5):

C. Tripartite Random Walk

The procedure of optimization in our task includes the objective functions O_1, O_2 and O_3 each of which are based on node samples which can be visited in the network to build context nodes and negative samples. In order to transform the structure of network into the skip-gram model, we follow metapath-guided random walks to traverse node paths in the tripartite graph to incorporate meaningful semantic relations in sequence of nodes.

Here we perform a biased and flexible truncated random walk by using the centrality measure calculated by HITS method [28] to control the number of random walk per node for learning implicit relations between same type of nodes. At each step, the transition from each node to the next one is dependant to the type of node. We highlight the core procedure given a heterogeneous graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ and the metapath scheme $\rho = v_0 \xrightarrow{R_0} v_1 \xrightarrow{R_1} v_2 \ldots \xrightarrow{R_t} v_t \xrightarrow{R_{t+1}} v_{t+1} \ldots \xrightarrow{R_n} v_{n+1}$. In each step, we select the next node using the following probability:

$$p(v_{i+1}|v_i, \rho) = \begin{cases} \dfrac{1}{|N_\rho^{(1)}(v_i)|} & \text{if } (v_{i+1}, v_i) \in N_\rho^{(1)}(v_i) \text{ and } \phi(v_{i+1}) = t+1 \\ 0 & \text{otherwise} \end{cases}$$
$$(8)$$

where $N_\rho^{(1)}(v_i)$ is the first order neighbor of the node v_i guided by the metapath of ρ, v_i is t-type node and v_{i+1} has the type of $t+1$ according to ρ. In tripartite network, there is no directed links within same type nodes, which means that the first order proximity between same type nodes is zero. Therefore, as it is shown in Figure 2, in the implicit relationship modeling, we generate a corpus of random walks guided by specified metapath schemes to incorporate the semantic information between nodes. Then, for each type of nodes, we check the random walks and filter out the nodes with different node types. In this case, we will have a sequence of nodes with same type to be used for skip-gram modeling. For each type nodes, we will optimize the loss functions O_1, O_2 and O_3.

D. Explicit Relationship Modeling

In previous sections, we discussed the approach to preserve the high order proximity within each set of node type through the metapath-guided random walk and the skip-gram model. However, the information conveyed by observed interconnecting edges in the tripartite graph is also important to consider. The weight of links show the tie strength and the local first order proximity between two nodes in the network. Therefore, in order to model these observed relations, given $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, the following joint probability between each pair of nodes (u_i, v_j), (v_j, w_k) and (u_i, w_k) can be defined where $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$, $u_i \in \mathcal{V}_1$, $v_j \in \mathcal{V}_2$, $w_k \in \mathcal{V}_3$ and $\mathcal{E}_1$ is the

500

subset of links in the graph between the pair of u_i and v_j, $\mathcal{E}_2$ denotes the edges starting from a node in $\mathcal{V}_2$ and ended to a node in $\mathcal{V}_3$ and $\mathcal{E}_3$ indicates the set of edges between pair of nodes from $\mathcal{V}_3$ and $\mathcal{V}_1$ node type sets respectively.

$$
\begin{aligned}
P_1(u_i, v_j) &= \frac{w_{ij}}{\sum_{e_{ij} \in \mathcal{E}_1} w_{ij}} \\
P_2(v_j, w_k) &= \frac{w_{jk}}{\sum_{e_{jk} \in \mathcal{E}_2} w_{jk}} \\
P_3(u_i, w_k) &= \frac{w_{ik}}{\sum_{e_{ik} \in \mathcal{E}_3} w_{ik}}
\end{aligned} \qquad (9)
$$

In this case, w_{ij} and w_{jk} are the weights of edge e_{ij} and e_{jk} between nodes in $\mathcal{V}_1$ and $\mathcal{V}_2$ or $\mathcal{V}_2$ and $\mathcal{V}_3$ parties respectively. To model these first order proximity information inspired by [28], we use the sigmoid function with the inner product kernel to estimate joint probability functions:

$$
\begin{aligned}
\hat{P}_1(u_i, v_j) &= \frac{1}{1 + e^{\theta_{u_i}^{\mathsf{T}} \theta_{\mathbf{v_j}}}} \\
\hat{P}_2(v_j, w_k) &= \frac{1}{1 + e^{\theta_{v_j}^{\mathsf{T}} \theta_{\mathbf{w_k}}}} \\
\hat{P}_3(u_i, w_k) &= \frac{1}{1 + e^{\theta_{u_i}^{\mathsf{T}} \theta_{\mathbf{w_k}}}}
\end{aligned} \qquad (10)
$$

where $\theta_{\mathbf{u_i}}$, $\theta_{\mathbf{v_j}}$ and $\theta_{\mathbf{w_k}}$ are the low-dimensional embedding vectors corresponding to nodes u_i, v_j and w_k in three types of nodes in the network. In order to preserve edge connection information, we define the objective function in form of KL-divergence [12] to minimize the difference between the distribution of edges and the reconstructed distribution:

$$
O_4 = KL(P_1 \| \hat{P}_1) = \alpha_1 - \sum_{e_{jk} \in \mathcal{E}_1} w_{jk} \log \hat{P}_1(u_i, v_j) \qquad (11)
$$

$$
O_5 = KL(P_2 \| \hat{P}_2) = \alpha_2 - \sum_{e_{jk} \in \mathcal{E}_2} w_{jk} \log \hat{P}_2(v_j, w_k) \qquad (12)
$$

$$
O_6 = KL(P_3 \| \hat{P}_3) = \alpha_2 - \sum_{e_{ik} \in \mathcal{E}_3} w_{ik} \log \hat{P}_3(u_i, w_k) \qquad (13)
$$

E. Model Training

To address implicit and explicit information in the tripartite network, we leverage the negative sampling strategy and metapath-guided random walks in addition to a reconstruction distribution function to define multiple objective functions. To preserve those information, the optimization step can be formed into a joint optimization framework through combining the objective functions as:

$$
\max \mathcal{L} = \alpha_1 O_1 + \alpha_2 O_2 + \alpha_3 O_3 + \beta_1 O_4 + \beta_2 O_5 + \beta_3 O_6 \qquad (14)
$$

where α_1, α_2, α_3, β_1, β_2, and β_3 are hyper parameters. For optimization, similar to other embedding methods like node2vec, we first create three corpora of vertex sequences through metapath-guided random walks per node type. For each node, we set up context nodes and negative nodes with the same node type in the sampled node sequence. Then we utilize Stochastic Gradient Descent algorithm to optimize the joint model. To calculate the gradient step to update the node

embedding and context vectors, we follow the formulations discussed in [28] to optimize the objective function. Therefore, the gradient of element $\beta_1 O_4$ can be calculated as follows. It is applied also for $\beta_1 O_5$ and $\beta_1 O_6$ elements respectively:

$$
\vec{u_i} = \vec{u_i} - \eta(\gamma w_{ij}(1 - \sigma(\vec{u}_i^T.\vec{v}_j)).\vec{v}_j) \qquad (15)
$$

$$
\vec{v_i} = \vec{v_i} - \eta(\gamma w_{ij}(1 - \sigma(\vec{u}_i^T.\vec{v}_j)).\vec{u}_j) \qquad (16)
$$

In the second step, we update node embedding and context vectors by calculating the gradient $\alpha_1 O_1$ element. We repeat this rule for two other objective function components; $\alpha_2 O_2$ and $\alpha_3 O_3$. The update rules are calculated as follows

$$
\vec{u_i} = \vec{v_i} - \eta \{ \sum_{\{v_c\} \cup N_S^{ns}(v_j)} \alpha_1 (I(z, u_i) - \sigma(\vec{u}_i^T.\vec{\theta}_z)).\vec{\theta}_z \} \qquad (17)
$$

$$
\vec{\theta}_z = \vec{\theta}_z - \eta \{ \alpha_1 (I(z, u_i) - \sigma(\vec{u}_i^T.\vec{\theta}_z)).\vec{u}_i \} \qquad (18)
$$

Where $I(z, u_i)$ is defined as an indicator function to specify that if node z is located in the context of node u_i or not. The pseudo code of training algorithm is listed in Algorithm 1.

F. TriNE Algorithm

Algorithm 1 lists the detailed procedure of the proposed tripartite network embedding method. It starts with initializing the node embedding vectors and context vectors. In the next step, the corpus of metapath-guided random walks are generated. They are split into three corpora with regard to node types to make homogeneous node sequences. Then, they are fed into the training procedure. The training iterations are repeated until the change between subsequent iterations become so small.

IV. EXPERIMENTS

A. Benchmark Datasets

In our experiments, we use two tripartite heterogeneous networks prepared from two real-world datasets including VisualizeUs [29] and MovieLens [29]. The former contains records from the tagging behavior of users on photos in which the weighted edges denote the number of times to have tagging of user on a specific image. The Latter is one of widely movie recommendation dataset consisting of tagging records of various users with regard to the list of movie items. The summary of some basic statistics of prepared datasets is listed in Table I.

B. Experimental Settings

As the common way to evaluate network embedding approaches, we choose the link prediction application for the evaluation task based on the intuition that the better representation of node in an embedding method can help us to predict links better in the network. Therefore, the link prediction can be cast as a binary classification problem to predict whether a user tags an image or a movie category. The link embedding vector samples are organized by calculating the average of embedding vectors of edge nodes. we randomly select one fifth of link samples as the test set while the remaining are the training set. We use the same embedding dimension as 128

Algorithm 1: TriNE: Tripartite Network Embedding Learning

Data: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$: a tripartite heterogeneous graph; metapath scheme set: $\rho = \{\rho_1, \rho_2, ...\}$, embedding dimension size: d, number of negative samples: ns, minimal and maximal number of random walk per node: $minT, MaxT$, Neighborhood size: k, random walk length: l

Result: The node embedding vectors: $\mathbf{U} \in \mathbb{R}^{|\mathcal{V}_1 \times d|}$, $\mathbf{P} \in \mathbb{R}^{|\mathcal{V}_2 \times d|}$, $\mathbf{C} \in \mathbb{R}^{|\mathcal{V}_3 \times d|}$

1 Initialize the embedding vectors in $\mathbf{U}$, $\mathbf{P}$ and $\mathbf{C}$;
2 Initialize the context vectors $\theta_{u_x}, \theta_{p_y}, \theta_{c_z}$ in $\mathbf{U}$, $\mathbf{P}$ and $\mathbf{C}$;
3 **for** $v \in \mathcal{V}$ **do**
4    $w = max(H(v), minT, maxT)$; where $H(v)$ is the HITS centrality measure;
5   **for** $l = 1, \cdots, w$ **do**
6   Cur=MetapathRandomWalk($\mathcal{G}, \rho, v, minT, maxT$) S+=Cur;
7   **end**
8 **end**
9 S_U, S_P, S_C=FilterNodeSequenceByType(S);
10
11 **for** $E = \mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ **do**
12   **for** *each edge* $e \in E$ **do**
13      $u, v \leftarrow e.u, e.v$;
14   update $\vec{u}$ and $\vec{v}$ by Eq.(15) and Eq.(16);
15      $S_1, S_2 = S_U, S_P$ if $e \in \mathcal{E}_1$ else S_P, S_C if $e \in \mathcal{E}_2$ else S_U, S_C;
16   **for** v_i, v_c *in the window of random walk* S_1 **do**
17   Do negative sampling;
18   Update $\vec{v_i}$ by Eq.(17);
19   Update $\vec{\theta_z}$ by Eq.(18) where $z \in \{v_c\} \cup N_S^{ns}(v_j)$;
20   **end**
21   **for** v_i, v_c *in the window of random walk* S_2 **do**
22   Do negative sampling;
23   Update $\vec{v_i}$ by Eq.(17);
24   Update $\vec{\theta_z}$ by Eq.(18) where $z \in \{v_c\} \cup N_S^{ns}(v_j)$;
25   **end**
26   **end**
27 **end**
28
29 **MetapathRandomWalk**($\mathcal{G}, \rho, v, minT, maxT$)
30 $s[0] \leftarrow v$;
31 **for** $i = 1, \cdots, l-1$ **do**
32   choose randomly u using Eq.(8);
33   s+=u;
34 **end**
35 return s;

Dataset	# of Users	# of Tags	# of Items	# of Edges
VisualizeUs	3,911	21,076	5,013	46,546
MovieLens	58,834	8,704	2,462	660,800

[14] and BiNE [28]. To evaluate the effectiveness of network embedding, we also consider the concatenation of embedding feature vectors of different methods for more comparisons. All experiments are evaluated based on the 5-fold cross validation. After getting a representation of network embedding methods for user type nodes as the target in training and test set, we train a binary classifier using a MLP neural network(3 layers with 100 neurons) on the training set to predict the link on the test set. We compare the result with different classifiers, including Support vector Machines (SVM) with RBF kernel and logistic regression (LR). We use the Area Under Receiver Operating Characteristics Curve (AUC-ROC) as the major evaluation metric because it shows the model accuracy of ranking positive cases versus negative ones. Besides, we also employ the area under precision-recall curve (AUC-PR) along with the F1 score as the additional performance metrics.

C. Baselines

We compare the performance of the proposed method TriNE with following baseline methods.

- **Metapath2vec:** The baseline representation learning method [14] for heterogeneous networks which applies metapath-guided random walks to model a heterogeneous skip-gram model.
- **BiNE:** The method [28] develops embedding vectors for bipartite networks. In our experiment, as the original datasets are in the form of tripartite networks, we execute this method on the subset of bipartite network including desired node types for the link prediction task.
- **Metapath2vec+BiNE:** In this experiment, we combine the embedding vectors calculated by metapath2vec method [14] with BiNE.
- **Metapath2vec+TriNE:** To evaluate the quality of features extracted from the proposed method, we concatenate the embedding vectors provided by the proposed method with those made by Metapath2vec method [14].

D. Performance Comparison

Table II reports the performance of different methods in two datasets.

TriNE outperforms the baselines In table II, we show that by applying the proposed embedding methods we can perform better than other approaches. In the case of using the MLP classifier, The proposed embedding method TriNE combined metapath2vec shows the best performance in almost all cases. It indicates the effectiveness of incorporating metapath semantic relations to learn node embedding vectors.

Hybrid embeddings is helpful. Comparing different methods, it can be seen that TriNE and also the hybrid method of

to make a fair comparison between methods. We compare our proposed method with two other methods, *i.e.*, metapath2vec

502

TABLE II
AVERAGE PERFORMANCE OF THE PROPOSED METHOD AND THE OTHER BASELINES ON TWO DATASETS

Classifier	Algorithm	Visualize-US			MovieLens		
		AUC-ROC	AUC-PR	F1	AUC-ROC	AUC-PR	F1
LR	Metapath2vec	0.5807	0.1811	0.2165	0.6045	0.2683	0.2903
	BiNE	0.6053	0.2016	0.2718	0.6687	0.2833	0.3061
	TriNE	0.6311	0.2243	0.3009	0.6795	0.3795	**0.4401**
	Metapath2vec+BiNE	**0.6675**	0.2569	**0.3494**	**0.7168**	0.3398	0.3799
	Metapath2vec+TriNE	0.6185	**0.2972**	0.3220	0.6793	**0.4208**	0.4339
MLP	Metapath2vec	0.7829	0.4398	0.4210	0.7366	0.3518	0.4052
	BiNE	0.6788	0.1942	0.2415	0.6994	0.3094	0.3939
	TriNE	0.7396	0.4314	0.4296	0.7146	0.3150	0.3475
	Metapath2vec+BiNE	**0.8424**	0.4353	0.3373	0.7417	0.3467	0.3299
	Metapath2vec+TriNE	0.8289	**0.6196**	**0.5254**	**0.7452**	**0.3920**	**0.4280**
SVM (RBF)	Metapath2vec	0.5631	0.3532	0.2153	0.5475	0.3776	0.1726
	BiNE	0.5113	0.1683	0.0442	0.5434	0.3170	0.1571
	TriNE	**0.6585**	**0.5319**	**0.4551**	0.5396	0.3909	0.1459
	Metapath2vec+BiNE	0.5657	0.3542	0.2208	**0.5632**	0.3951	**0.2206**
	Metapath2vec+TriNE	0.5575	0.3624	0.1957	0.5515	**0.4215**	0.1870

metapath2vec and TriNE generally outperform the all baseline methods. We show that applying heterogeneous skip-gram models for all node types is more effective method for the link prediction. The variant of TriNE to combine embedded features with metapath2vec method outperform the remaining methods based on the majority of metrics. It presents the usefulness of metapath-guided random walks than typical random walks to capture the discriminative information for the link prediction.

V. CONCLUSION

In this paper, we proposed a tripartite network embedding learning method to model explicit relationships between nodes (observed links between nodes), and also capture implicit relationships between tripartite nodes (unobserved links across tripartite node sets). We applied a joint optimization to train a heterogeneous skip-gram model to capture semantic and structural relations collected by utilizing metapath-guided random walks. We validated the performance of the proposed method on real-world data gathered for the link prediction task in the movie and image tagging. The results showed the effectiveness of the proposed method.

In our study, we applied a linear combination of multiple loss terms as the objective function to jointly model explicit and implicit relations in the tripartite heterogeneous graph for the link prediction task. In our future work, we are seeking to investigate a non-linear combination for the network representation learning in the other applicable tasks.

REFERENCES

[1] K. Zhang, S. Bhattacharyya, and S. Ram, "Large-scale network analysis for online social brand advertising," *MIS Quarterly*, vol. 40, no. 4, pp. 849–868, 2016.

[2] M. Takarabe, D. Shigemizu, M. Kotera, S. Goto, and M. Kanehisa, "Network-based analysis and characterization of adverse drug–drug interactions," *Journal of Chemical Information and Modeling*, vol. 51, no. 11, pp. 2977–2985, 2011, pMID: 21942936. [Online]. Available: https://doi.org/10.1021/ci200367w

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

[4] Y. Qu, C. H. G. Bohui Fang et.al. Shanghai Jiao Tong University, Y. Yu, and X. He, "Product-based neural networks for user response prediction over multi-field categorical data," *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 1, 2019.

[5] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network Representation Learning: A Survey," *IEEE Trans. on Big Data*, vol. 6, p. 3–29, 2020.

[6] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000. [Online]. Available: http://www.sciencemag.org/cgi/content/abstract/290/5500/2323

[7] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, p. 2319, 2000.

[8] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[9] B. N. Araabi and Z. Gharibshah, "Face recognition with manifold-based kernel discriminant analysis," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–8.

[10] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Info. Proc. Systems 26*, 2013, pp. 3111–3119.

[12] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," *ArXiv*, vol. abs/1503.03578, 2015.

[13] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks," in *WSDM '17*, 2017.

[14] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD '17*, 2017.

[15] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. of the 20th ACM SIGKDD Conference*. ACM, 2014, pp. 701–710.

[16] C. Zhang, A. Swami, and N. V. Chawla, "Shne: Representation learning for semantic-associated heterogeneous networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, ser. WSDM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 690–698. [Online]. Available: https://doi.org/10.1145/3289600.3291001

[17] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, and Y. Li, "Metapath-guided heterogeneous graph neural network for intent recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 2478–2486. [Online]. Available: https://doi.org/10.1145/3292500.3330673

[18] H. Wang, F. Zhang, M. Zhang *et al.*, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," *Proc of KDD*, p. 968–977, 2019.

[19] H. Wang, F. Zhang, X. Xie *et al.*, "Dkn: Deep knowledge-aware network for news recommendation," *ArXiv*, vol. abs/1801.08284, 2018.

[20] Y. Qu, T. Bai, W. Zhang *et al.*, "An end-to-end neighborhood-based interaction model for knowledge-enhanced recommendation," *ArXiv*, vol. abs/1908.04032, 2019.

[21] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ser. CIKM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 417–426. [Online]. Available: https://doi.org/10.1145/3269206.3271739

[22] G. Zhou, C. Song, X. Zhu *et al.*, "Deep interest network for click-through rate prediction," *Proc of KDD*, p. 1059–1068, 2018.

[23] Z. Gharibshah, X. Zhu *et al.*, "Deep learning for user interest and response prediction in online display advertising," *Data Science and Engineering*, Jan 2020.

[24] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous Information Network Embedding for Recommendation," *arXiv e-prints*, p. arXiv:1711.10730, Nov. 2017.

[25] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 635–644. [Online]. Available: https://doi.org/10.1145/3097983.3098063

[26] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, ser. WSDM '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 283–292. [Online]. Available: https://doi.org/10.1145/2556195.2556259

[27] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "MetaGraph2Vec: Complex Semantic Path Augmented Heterogeneous Network Embedding," *The 22nd Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD)*, p. arXiv:1803.02533, Mar. 2018.

[28] M. Gao, L. Chen, X. He, and A. Zhou, "Bine: Bipartite network embedding," in *Proc. of the ACM SIGIR Conference*, 2018.

[29] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.