Noname manuscript No.
(will be inserted by the editor)

Multi-task Network Embedding

Linchuan Xu [1] · Xiaokai Wei[2] · Jiannong Cao[1] · Philip S. Yu[3]

Received: date / Accepted: date

Abstract As there are various data mining applications involving network analysis, network embedding is frequently employed to learn latent representations or embeddings that encode the network structure. However, existing network embedding models are only designed for a single network scenario. It is common that nodes can have multiple types of relationships in big data era, which results in multiple networks, e.g., multiple social networks and multiple gene regulatory networks. Jointly embedding multiple networks thus may make network-specific embeddings more comprehensive and complete as the same node may expose similar or complementary characteristics in different networks. In this paper, we thus propose an idea of multi-task network embedding to jointly learn multiple network-specific embeddings for each node via enforcing an extra information-sharing embedding. We instantiate the idea in two types of models that are different in the mechanism for enforcing the information-sharing embedding. The first type enforces the information-sharing embedding as a common embedding shared by all tasks, which is similar to the concept of the common metric in multi-task metric learning while the second type enforces the information-sharing embedding as a consensus embedding on which all network-specific embeddings agree. Moreover, we propose two mechanisms for embedding the network structure, which are first-order proximity preserving and second-order proximity preserving. We demonstrate through comprehensive experiments on three real-world datasets that the proposed models outperform recent network embedding models in applications including visualization, link prediction, and multi-label classification.

Keywords Network embedding · Representation learning · Multi-task learning · Data mining

This paper is an extension version of the DSAA'2017 Research Track paper titled "Multi-task Network Embedding"[32].

✉Linchuan Xu
E-mail: 13901626r@connect.polyu.hk

Xiaokai Wei
E-mail: weixiaokai@gmail.com

Jiannong Cao
E-mail: csjcao@comp.polyu.edu.hk

Philip S. Yu
E-mail: psyu@uic.edu

[1]Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

[2]Facebook Inc., 1 Hacker Way, Menlo Park, CA, USA

[3]Department of Computer Science, University of Illinois at Chicago, Chicago, Illinois, USA

1 Introduction

Entities can have various kinds of explicit or implicit interactions, such as friendships, co-authorships and co-concurrence, which result in various kinds of networks. Consequently, there are lots of data mining applications involving network analysis, such as link prediction [14] [28], network visualization [15], information diffusion [3], and node classification [7]. Since traditional tuple-based data mining models, such as SVM and Logistic Regression, cannot be directly applied to networked datasets, network embedding is frequently employed to learn latent representations or embeddings for network nodes. Network embedding learns node embeddings basically by preserving the network structure [21, 25, 13, 34, 33, 30, 29, 37]. However, most existing

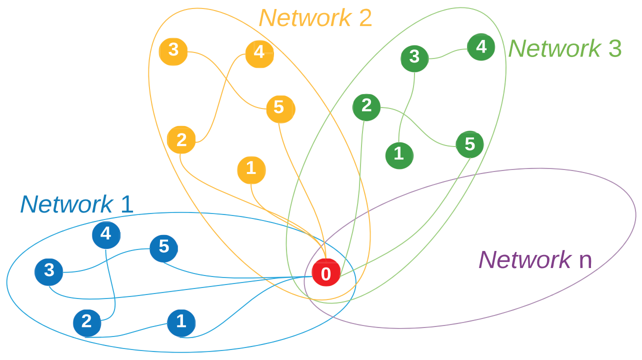

Fig. 1: A toy example of a node participating in multiple undirected networks, which is drawn in the form of the ego network from the perspective of node 0.

methods are only applicable to a single network scenario.

In practice, a node can participate in multiple networks simultaneously as illustrated in Fig. 1, e.g.,

– Genes can participate in different gene regulatory networks where genes have different kinds of interactions, such as protein-protein interaction (PPI) and health or disease related interactions.
– Researchers can form co-authorship networks as well as citation networks.
– People can join multiple online social networks, such as Facebook, Twitter, and LinkedIn.

A node may expose similar characteristics in different networks, e.g., node 0 is connected to node 5 in all the networks. A real case can be that a user is connected to another user in all the online social networks they participate in. Besides, a node may expose complementary characteristics in different networks, e.g., node 0 is not connected to node 1 in network 3 but is connected to node 1 in both network 1 and network 2. One thus can leverage these similar and complementary characteristics of a particular node exposed in all the networks to make every single network-specific embedding more comprehensive and complete.

In this paper, we thus propose an idea of multi-task network embedding (MTNE) to jointly learn multiple network-specific embeddings for each node via enforcing an extra information-sharing embedding for the first time. In this paper, a task is referred to as embedding a single network, or learning a network-specific embedding for each node. The information-sharing embedding is responsible for sharing the node characteristics exhibited in one network to another as its name suggests. The proposed idea of multi-task network embedding is instantiated in two types of models, which are different in the mechanism for enforcing the information-sharing embedding.

The first type enforces the information-sharing embedding as a common embedding, which is similar to the concept of the common metric in multi-task metric learning [20]. The embedding is named as common embedding because it is shared by all embedding tasks. In this way, the common embedding is responsible for capturing the commonalities shared by all networks while each network-specific embedding only captures the characteristics which are specific to a particular network. The idea of shared knowledge in multi-task learning has been proved very effective in many multi-task learning models, such as multi-task SVM [12], multi-task deep learning [9], and multi-task distance metric learning [20]. Because both the common embedding and network-specific embeddings only capture partial node characteristics, they are combined to be a complete embedding in each task.

The second type enforces the information-sharing embedding as a consensus embedding on which all network-specific embeddings should agree. The intuition behind this mechanism is that different network-specific characteristics are viewed as different expressions of the same set of node characteristics. In other words, different network-specific embeddings actually depict the same node but from different perspectives. For example, all network-specific embeddings of node 0 in Fig. 1 capture the characteristics of node 0 but under different scenarios, such as different social environment. Since all the network-specific embeddings agree on the consensus embedding, the commonalities can be fused on the consensus embedding as well as imported into network-specific embeddings through the consensus embedding. To make network-specific embeddings agree on the consensus embedding, we regularize network-specific embeddings to be similar to the consensus embedding.

Moreover, to demonstrate the effectiveness of the two information-sharing embeddings, we propose two methods for embedding the network structure, which are different in the order of node proximity that is preserved in the embedding space. The first method preserves the first-order node proximity while the second method preserves the second-order node proximity. These two orders of proximities are studied because it has been shown that the network structure can be well embedded when the first-order or the second-order node proximity is preserved [25].

To sum up, we propose two types of mechanisms for enforcing the information-sharing embedding, and two types of mechanisms for embedding the network structure. Different mechanisms for enforcing the information-sharing embedding are proposed because they have different advantages, which are presented in the following sections. Different mechanisms for embedding the network structure are proposed because we would like to demonstrate the effectiveness of the information-sharing embedding regardless of the method for embedding the network structure.

The contributions of this paper are summarized as follows:

– To the best of our knowledge, this is the first work to jointly embed multiple networks where the multiple networks share the same set of nodes but may have different connections among the nodes.
– To the best of our knowledge, this is the first work to apply the idea of shared knowledge in multi-task learning into network embedding, which is demonstrated by enforcing the information-sharing embedding as a common embedding shared by all tasks for embedding a single network.
– We propose four models for multi-task network embedding corresponding to two types of mechanisms for enforcing the information-sharing embedding and two types of mechanisms for embedding the network structure.
– We conduct comprehensive experiments on three real-world datasets to demonstrate that the proposed models outperform recent network embedding models in applications including visualization, link prediction and multi-label classification.
– This work shows jointly embedding multiple networks via enforcing an information-sharing embedding is an effective way to fuse information in multiple networks.

The rest of the paper is organized as follows. Section 2 presents related work. We present the proposed models through section 3 to section 6. Section 7 presents empirical evaluation. In section 8, we summarize and introduce future work.

2 Related Work

This paper deals with leaning low-dimensional representations of nodes through embedding the network structure. Besides, to jointly model multiple networks which share the same set of nodes, this paper proposes to learn a shared pool of knowledge. As a result, network embedding and multi-task learning are the two major categories of related work.

2.1 Network Embedding

Previously, network embedding [10, 4] has been proposed to reduce the dimensionality of features of independent data points because high-dimensional features make it inefficient or even infeasible to train data mining models. In this scenario, they have to construct a network before performing the embedding, where typical networks are nearest neighbor networks.

Recently, network embedding methods under the natural network scenario have been studied. Many methods are based on Skip-gram [17], a language model, such as DeepWalk [21], node2vec [13], and metapath2vec [11]. The basic idea is to generate sequences of nodes via random walks and then enforce sequences of nodes to be close in the embedding space of interest through a specially designed neural network. With DeepWalk being proved to be equivalent to matrix factorization, TADW [38] embeds both the network structure and node content via matrix factorization.

There are also many other methods. LINE [25] presents pairs of nodes with first-order links or second-order links to be close in the embedding space. Besides simply preserving the network structure, some methods also consider network properties, such as HOPE [19] preserving asymmetric transitivities, M-NMF [27] preserving communities, and MSRE [35] preserving social roles. EOE [34] jointly embeds two networks which are inter-connected. But the two networks do not share the same set of nodes. ICANE [36] embeds networks with edge content. DLP [31] embeds signed network especially for link prediction. This paper, instead, jointly embeds multiple networks which cannot be handled by aforementioned methods.

2.2 Multi-task Learning

The idea of multi-task learning [2, 9] is to jointly learn multiple task-specific knowledge via enforcing a pool of shared knowledge, such as a shared parameterization or representation, such that tasks can benefit from each other. There are also many multi-task learning studies on networks, mostly for link prediction [40, 39] and clustering [22, 18]. In this way, these methods are application-specific. Instead, the proposed MTNE methods are designed to embed network structure but not for any specific applications.

We have to mention that one of the proposed ways to enforce the information-sharing embedding, i.e., the common embedding, is similar to the shared knowledge learning in multi-task SVM [12] and multi-task metric learning [20]. But we further propose another way to incorporate the information-sharing embedding, i.e., enforcing it as a consensus embedding on which all network-specific embeddings should agree.

3 Overview

The studied multiple networks are defined as follows:

DEFINITION 1. A collection of multiple networks is defined as $G = \{G^1, ..., G^t, ..., G^T\}$, where T is the number of networks, and G^t refers to $G^t(V, E^t)$ with V representing the set of nodes summarized from the all networks and E^t representing a set of network-specific weighted or unweighted, directed or undirected edges.

It is worthy of noting that a particular network may not contain every single node of V because some nodes having interactions in one network may not have interactions in another network. But we can still assume that the each network contains all the nodes because it is possible that two nodes may interact in all the networks of interest. Because the same nodes are likely to expose similar characteristics in different networks, this assumption actually can help solve the cold-start problem for a particular network that does not contain all the nodes. But the cold-start problem is not explicitly addressed in this paper.

The problem is stated as follows:

DEFINITION 2. Given multiple networks G, the objective is to jointly learn multiple network-specific low-dimensional and continuous node embeddings where each network-specific embedding preserves the corresponding network structure as well as information from other networks.

The proposed idea of multi-task network embedding jointly learns multiple-network embeddings via enforcing an information-sharing embedding for each node. The idea is instantiated in two mechanisms for enforcing the information-sharing embedding. The first mechanism enforces the information-sharing embedding as a common embedding shared by all network-specific embeddings while the second mechanism enforces the information sharing embedding as a consensus embedding on which all network-specific embeddings agree. Since the proposed mechanisms for enforcing the information-sharing embedding do not require a particular method for embedding the network structure, we propose two network embedding methods to better illustrate the use of information-sharing embedding, i.e., first-order proximity preserving and second-order proximity preserving. Hence, there are fours models proposed in this paper. While organizing the four models, we first present two models based on the first-order proximity preserving and then the other two models based on the second-order proximity preserving.

4 MTNE via First-order Proximity Preserving (FOPP)

In this section, we present the two models that embed the network structure via preserving first-order node proximity in the embedding space. Specifically, we first present the method for embedding a single network, and then extend it to the two methods for jointly embedding multiple networks.

4.1 Single Network Embedding via FOPP

First-order proximity preserving means directly preserving the relationships of two nodes without considering all the other nodes, i.e., presenting pairs of nodes to be close in the embedding space if the pairs are connected in the network or to be apart if they are not connected. We define the first-order closeness of two nodes in the embedding space of interest as follows:

DEFINITION 3. First-order closeness directly measures Euclidean distance of two embeddings, which is quantified as follows:

$$p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t) = \frac{1}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}}, \tag{1}$$

where $\boldsymbol{v}_i^t \in \mathbb{R}^L$ and $\boldsymbol{v}_j^t \in \mathbb{R}^L$ are the embeddings of node i and j for network t, respectively, and L is the dimension of embeddings. All embeddings are denoted in column vector.

The closeness can be defined in this way because Eq. (1) is positively correlated with the inner product of two vectors, and the inner product is a measure of closeness of two points in Euclidean space. Moreover, the closeness is essentially the probability that there exists an edge between two nodes, which can be employed to formulate the optimization objective presented as follows.

Formally, the proposed first-order proximity preserving applied to a single network is formulated into an optimization problem according to the maximum likelihood estimation as follows:

$$\max_{\boldsymbol{V}^t} \prod_{(i,j)\in E^t, (h,k)\notin E^t} p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t)(1 - p(\boldsymbol{v}_h^t, \boldsymbol{v}_k^t)) \tag{2}$$

The difference between the denotation of $p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t)$ and $p(\boldsymbol{v}_h^t, \boldsymbol{v}_k^t)$ is that node i and node j belong to a pair of nodes connected by an edge while node h and node k belong to a pair of nodes not connected in G^t. Hence, Eq. (2) maximizes both the probabilities of existing edges and those of non-existing edges. The consideration of non-existing edges is important because it presents nodes not connected to be away from each other in the embedding space.

The multiplication maximization problem is usually transformed to an equivalent minimization problem by taking negative natural logarithm, which is denoted as follows:

$$\min_{\boldsymbol{V}^t} l^{FOPP}, \tag{3}$$

where

$$\begin{aligned} l^{FOPP} = &- \sum_{(i,j)\in E^t} w_{ij}^t \log p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t) \\ &- \sum_{(h,k)\notin E^t} \log(1 - p(\boldsymbol{v}_h^t, \boldsymbol{v}_k^t)), \end{aligned} \tag{4}$$

where $w_{ij}^t \in \mathbb{R}$ is the weight of the edge (i, j) to reflect to relationship strength, and does not violate the original objective as indicated in Eq. (2).

4.2 Multi-task Metric Learning

Before extending the method for single network embedding to multiple network embedding via enforcing the information-sharing embedding as a common embedding, we briefly review the multi-task distance metric learning [20] which also introduces a common metric shared by all tasks.

In multi-task distance metric learning, the objective is to jointly learn multiple task-specific distance metrics for measuring the distance among relational data points so that the task-specific distance among data points of the same category is small while the distance among data points of different categories is large. The distance measure is Mahalanobis distance in this setting. The multiple tasks correspond to multiple sets of categories, or multiple sets of labels. One assumption about the model is that multiple tasks are different but related, which lays the foundation of learning a pool of shared knowledge for multiple tasks. To capture and utilize the shared knowledge, the model enforces a common metric shared by all the tasks.

Formally, the Mahalanobis distance for task t with the common metric is quantified as follows:

$$d_t(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T(\boldsymbol{M}_t + \boldsymbol{M}_0)(\boldsymbol{x}_i - \boldsymbol{x}_j)}, \tag{5}$$

where $\boldsymbol{x}_i \in \mathbb{R}^L$ and $\boldsymbol{x}_j \in \mathbb{R}^L$ are two data points denoted in column vector, L is the dimension, $\boldsymbol{M}_t \in \mathbb{R}^{L\times L}$ is a Mahalanobis metric for task t, and $\boldsymbol{M}_0 \in \mathbb{R}^{L\times L}$ is the common metric shared by all tasks. Since all the tasks are different but related, the task-specific metric is expected to capture task-specific knowledge while the common metric is expected to capture common knowledge shared by all the tasks. This elegant formulation has been proved very effective in the multi-task metric learning [20].

4.3 FOPP with Common Embedding

This section extends the proposed single-task network embedding via FOPP to multi-task network embedding. The extension is mainly performed by enforcing a common embedding for each node, which is similar to the concept of the common metric in multi-task metric learning. The common embedding is defined as follows:

DEFINITION 4. The **common embedding** for node i is denoted as $\boldsymbol{v}_i^0 \in \mathbb{R}^L$ where the superscript of 0 means that it is not specific to any network.

With the common embedding, the edge probability for network t in the multi-task setting is quantified as follows:

$$p(\boldsymbol{v}_i^t, \boldsymbol{v}_i^0, \boldsymbol{v}_j^t, \boldsymbol{v}_j^0) = \frac{1}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top(\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}} \tag{6}$$

In this way, the common embedding and a network-specific embedding are combined to be a complete embedding for the closeness measurement. A different view is that the embedding of each node for each task is separated into a common embedding and a network-specific embedding. Because the common embedding is shared in all network-specific closeness measurement, it is expected to capture the common characteristics of the node across all the networks. In this way, the network-specific embeddings can be more comprehensive as it utilizes information from other networks. Because the network structure is embedded via FOPP and the common embedding is shared by all networks, the method is shorted as FOPPS where "S" is for sharing.

By following the way of formulating the optimization objective in the single-task network embedding, the loss function for jointly learning multi-task embeddings is quantified as follows:

$$\begin{aligned} l^{FOPPS} = &\sum_t \Bigg[- \sum_{(i,j)\in E^t} w_{ij}^t \log p(\boldsymbol{v}_i^t, \boldsymbol{v}_i^0, \boldsymbol{v}_j^t, \boldsymbol{v}_j^0) \\ &- \sum_{(h,k)\notin E^t} \log(1 - p(\boldsymbol{v}_h^t, \boldsymbol{v}_h^0, \boldsymbol{v}_k^t, \boldsymbol{v}_k^0))\Bigg] \\ &+ \lambda \sum_t ||\boldsymbol{V}^t||_F^2 + \lambda ||\boldsymbol{V}^0||_F^2, \end{aligned} \tag{7}$$

where $\boldsymbol{V}^t \in \mathbb{R}^{L\times N}$ and $\boldsymbol{V}^0 \in \mathbb{R}^{L\times N}$ are matrix denotations of network-specific embeddings and the common embeddings, respectively, N is the number of nodes, $\lambda \in \mathbb{R}$ and $\lambda \in \mathbb{R}$ are regularization coefficients, and $|| \cdot ||_F^2$ is the square of Frobenius norm.

4.4 FOPP with Consensus Embedding

This section extends the proposed single network embedding to multi-network embedding by enforcing the information-sharing embedding as a consensus embedding, which is defined as follows:

DEFINITION 5. The **consensus embedding** for node i is denoted as $\boldsymbol{z}_i \in \mathbb{R}^L$ on which all network-specific embeddings of node i should agree.

As mentioned in the introduction, network-specific embeddings of a particular node can be considered as different expressions of the same node characteristics under different circumstances. Hence, they all should agree on the same node characteristics, which are expected to be captured in the consensus embedding. To make network-specific embeddings comply with the consensus embedding, they are regularized to be similar to the consensus embedding, which can be achieved by the following formula:

$$\min_{\boldsymbol{V}^t} ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2. \tag{8}$$

where $\boldsymbol{Z} \in \mathbb{R}^{L \times N}$. Eq. (8) minimizes the difference between network-specific embeddings and the consensus embedding. The model enforcing the consensus embedding is shorted as FOPPR where "R" is for regularization.

To jointly embed multiple networks, the loss function can be a direct summation of network-specific losses and the regularization loss, which is quantified as follows:

$$l^{FOPPR} = \sum_t l^{FOPP} + \sum_{t=1} ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 + \lambda \sum_t ||\boldsymbol{V}^t||_F^2 + \lambda ||\boldsymbol{Z}||_F^2, \tag{9}$$

where l^{FOPP} is quantified in Eq. (4). The summation is reasonable in that network-specific embeddings should both encode network-specific characteristics and agree on the consensus embedding. We leave more sophisticated ways for the joint embedding as future work. It is worthy of noting that the consensus embedding is also to be learned by minimizing the regularization loss as indicated in Eq. (8) but with respect to $\boldsymbol{Z}$. As a result, the consensus embedding summarizes all network-specific information. When it comes to regularizing network-specific embeddings to be similar to the consensus embedding, network-specific information can be shared among different tasks through the consensus embedding.

5 MTNE via Second-order Proximity Preserving (SOPP)

This section proposes another two models for the multi-task network embedding based on the second-order proximity preserving (SOPP). Similarly, we first present the mechanism of SOPP for embedding a single network, then extends it to multiple networks with the aforementioned two mechanisms for enforcing the information-sharing embedding.

5.1 Single Network Embedding via SOPP

The second-order proximity preserving is similar to the first-order proximity preserving except that the relationships of two nodes are indirectly preserved through other nodes that are connected to the two nodes. More specifically, two nodes are expected to be similar when they share similar connections to other nodes. In this case, other nodes are treated as contexts of the two nodes of interest [25]. Hence, there are two embeddings for each node, i.e., one embedding as node and the other embedding as context. Then to preserve the second-order proximity, in the embedding space, nodes are presented to be close to contexts that are connected to them and presented to be away from those contexts that are not connected to them.

Formally, the second-order closeness is defined as follows:

DEFINITION 6. Second-order closeness measures the Euclidean distance between a node embedding and a context embedding, which is quantified as follows:

$$p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t) = \frac{1}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top \boldsymbol{v}_i^t\}}, \tag{10}$$

where $\boldsymbol{c}_j^t$ is the embedding of node i as context for network t.

Also, the optimization objective of SOPP for embedding a network can be formulated through the maximum likelihood estimation principle. And the corresponding equivalent minimization objective can be obtained as follows:

$$\begin{aligned} l^{SOPP} = &- \sum_{(i,j) \in E^t} w_{ij}^t \log p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t) \\ &- \sum_{(h,k) \notin E^t} \log(1 - p(\boldsymbol{c}_k^t, \boldsymbol{v}_h^t)), \end{aligned} \tag{11}$$

5.2 SOPP with Common Embedding

Similar to extend the single network embedding via FOPP to the multi-network embedding, a common node embedding is added to each network-specific node embedding to be a complete embedding. Note that we do not introduce a common context embedding because context embeddings are not our final targets. Hence, the optimization objective for jointly embedding multiple networks by following Eq. (7) can be obtained as follows:

$$
\begin{aligned}
l^{SOPPS} =& \sum_t \Bigg[- \sum_{(i,j)\in E^t} w_{ij}^t \log p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t, \boldsymbol{v}_i^0) \\
& - \sum_{(h,k)\notin E^t} \log(1 - p(\boldsymbol{c}_k^t, \boldsymbol{v}_h^t, \boldsymbol{v}_h^0)) \Bigg] \qquad (12) \\
& + \lambda \sum_t (||\boldsymbol{C}^t||_F^2 + ||\boldsymbol{V}^t||_F^2) + \lambda ||\boldsymbol{V}^0||_F^2,
\end{aligned}
$$

where $p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t, \boldsymbol{v}_i^0)$ is quantified as follows:

$$
p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t, \boldsymbol{v}_i^0) = \frac{1}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}}, \qquad (13)
$$

This model is referred to as SOPPS where "S" is for "sharing" in the rest of the paper.

5.3 SOPP with Consensus Embedding

It is also straightforward to extend from Eq. (11) to jointly embedding multiple networks by introducing a consensus embedding for each node. The optimization objective is thus obtained as follows:

$$
\begin{aligned}
l^{SOPPR} =& \sum_t \Bigg[- \sum_{(i,j)\in E^t} w_{ij}^t \log p(\boldsymbol{c}_j^t, \boldsymbol{v}_i^t) \\
& - \sum_{(h,k)\notin E^t} \log(1 - p(\boldsymbol{c}_k^t, \boldsymbol{v}_h^t)) \Bigg] \\
& + \sum_t ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 + \lambda \sum_t ||\boldsymbol{V}^t||_F^2 + \lambda ||\boldsymbol{V}^0||_F^2,
\end{aligned}
\qquad (14)
$$

Correspondingly, this model is referred to as SOPPR where "R" is for regularization.

6 The Optimization

All the four methods proposed above require solving minimization problems. In this section, we thus first present derivatives for minimizing the loss functions. Based on the derivatives, we then present the proposed optimization algorithm

6.1 Derivatives for FOPPS

It is worthy of noting that the loss function l^{FOPPS} involves more than one variable. We thus compute partial derivatives with respect to (w.r.t.) each variable.

The partial derivative w.r.t. $\boldsymbol{v}_i^t$ is computed as follows: $\frac{\partial l^{FOPPS}}{\partial \boldsymbol{v}_i^t} =$

$$
\begin{aligned}
& - \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}} \times (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0) \right] \\
& + \sum_{(i,k)\notin E^t} \left[\frac{(\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)\}} \right] \\
& + 2\lambda \boldsymbol{v}_i^t
\end{aligned}
\qquad (15)
$$

The partial derivative w.r.t. $\boldsymbol{v}_i^0$ is computed as follows: $\frac{\partial l^{FOPPS}}{\partial \boldsymbol{v}_i^0} =$

$$
\begin{aligned}
& \sum_d \Bigg[- \sum \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}} \times (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0) \right] \\
& + \sum \left[\frac{(\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)\}} \right] \Bigg] \\
& + 2\lambda \boldsymbol{v}_i^0,
\end{aligned}
\qquad (16)
$$

where the two subscriptions about the edges, i.e., $(i,j) \in E^t$ and $(i,k) \notin E^t$, are omit due to space consideration.

6.2 Derivatives for FOPPR

Similarly, l^{FOPPR} involves more than one variable. To derive the derivative with respect to a particular network-specific embedding, l^{FOPPR} reduces to the following function:

$$
l^{FOPPR} = l^{FOPP} + ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 \qquad (17)
$$

The partial derivative w.r.t. to $\boldsymbol{v}_i^t$ thus can be obtained as follows:

$$
\frac{\partial l^R}{\partial \boldsymbol{v}_i^t} = \frac{\partial l^{FOPP}}{\partial \boldsymbol{v}_i^t} + 2(\boldsymbol{v}_i^t - \boldsymbol{z}_i), \qquad (18)
$$

where

$$
\begin{aligned}
\frac{\partial l^{FOPP}}{\partial \boldsymbol{v}_i^t} =& - \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}} \times \boldsymbol{v}_j^t \right] \\
& + \sum_{(i,k)\notin E^t} \left[\frac{\boldsymbol{v}_k^t}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_k^t\}} \right] \qquad (19) \\
& + 2\lambda \boldsymbol{v}_i^t,
\end{aligned}
$$

With respect to $\boldsymbol{Z}$, the minimization problem reduces to the following one:

$$\min_{\boldsymbol{Z}} \sum_t ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 + \lambda ||\boldsymbol{Z}||_F^2, \tag{20}$$

which is a convex problem, and it is easy to obtain the optimal $\boldsymbol{Z}$ by setting its derivative to zero, where the derivative is obtained as follows:

$$\sum_t 2(\boldsymbol{Z} - \boldsymbol{V}^t) + 2\lambda \boldsymbol{Z} \tag{21}$$

The optimal $\boldsymbol{Z}$ is thus quantified as follows:

$$\boldsymbol{Z} = \frac{\sum_t \boldsymbol{V}^t}{T + \lambda} \tag{22}$$

6.3 Derivatives for SOPPS

The partial derivative of l^{SOPPS} with respect to v_i^t is obtained as follows: $\frac{\partial l^{FOPPS}}{\partial \boldsymbol{v}_i^t} =$

$$\begin{aligned} &- \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}} \times \boldsymbol{c}_j^t \right] \\ &+ \sum_{(i,k)\notin E^t} \left[\frac{\boldsymbol{c}_k^t}{1 + \exp\{-(\boldsymbol{c}_k^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}} \right] + 2\lambda \boldsymbol{v}_i^t \end{aligned} \tag{23}$$

The partial derivative of l^{SOPPS} with respect to v_i^0 is obtained as follows: $\frac{\partial l^{FOPPS}}{\partial \boldsymbol{v}_i^0} =$

$$\begin{aligned} &\sum_t \Bigg[- \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}} \times \boldsymbol{c}_j^t \right] \\ &+ \sum_{(i,k)\notin E^t} \left[\frac{\boldsymbol{c}_k^t}{1 + \exp\{-(\boldsymbol{c}_k^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}} \right] \Bigg] + 2\lambda \boldsymbol{v}_i^0 \end{aligned} \tag{24}$$

The partial derivative of l^{SOPPS} with respect to c_j^t is obtained as follows: $\frac{\partial l^{FOPPS}}{\partial \boldsymbol{c}_j^t} =$

$$\begin{aligned} &- \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)\}} \times (\boldsymbol{v}_i^t + \boldsymbol{v}_i^0) \right] \\ &+ \sum_{(j,h)\notin E^t} \left[\frac{\boldsymbol{v}_h^t + \boldsymbol{v}_h^0}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top (\boldsymbol{v}_h^t + \boldsymbol{v}_h^0)\}} \right] + 2\lambda \boldsymbol{c}_j^t \end{aligned} \tag{25}$$

6.4 Derivatives for SOPPR

The partial derivative of l^{SOPPR} with respect to v_i^t is obtained as follows:

$$\frac{\partial l^{SOPPR}}{\partial \boldsymbol{v}_i^t} = \frac{\partial l^{SOPP}}{\partial \boldsymbol{v}_i^t} + 2(\boldsymbol{v}_i^t - \boldsymbol{z}_i), \tag{26}$$

Algorithm 1: The alternating optimization algorithm

```
Input  : $G = \{G^1, ..., G^t, ..., G^T\}$, $L$, and $r$
Output: $\boldsymbol{V}^1, ..., \boldsymbol{V}^t, ..., \boldsymbol{V}^T$, and $\boldsymbol{V}^0$ or $\boldsymbol{Z}$

Perform single-task network embedding to pre-train
  $\boldsymbol{V}^1, ..., \boldsymbol{V}^t, ..., \boldsymbol{V}^T$ and $\boldsymbol{C}^1, ..., \boldsymbol{C}^t, ..., \boldsymbol{C}^T$;
repeat
    Solve $\boldsymbol{V}^0$ for FOPPS and SOPPS with gradient
      descent, or find the optimal $\boldsymbol{Z}$ with Eq. (22) for
      FOPPR and SOPPR while fixing all the other
      variables;
    for $t \leftarrow$ 1 to $T$ do
        Solve $\boldsymbol{V}^t$ for all the four models with gradient
          descent while fixing all the other variables;
        Specifically for FOPPS and SOPPS, solve $\boldsymbol{C}^t$
          with gradient descent while fixing all the
          other variables;
until converge or maximum iteration;
return $\boldsymbol{V}^1, ..., \boldsymbol{V}^t, ..., \boldsymbol{V}^T$, and $\boldsymbol{V}^0$ or $\boldsymbol{Z}$
```

where $\frac{\partial l^{SOPP}}{\partial \boldsymbol{v}_i^t}$ is quantified as follows:

$$\begin{aligned} \frac{\partial l^{SOPP}}{\partial \boldsymbol{v}_i^t} = &- \sum_{(i,j)\in E^t} \left[\frac{w_{ij}^t \exp\{-(\boldsymbol{c}_j^t)^\top \boldsymbol{v}_i^t\}}{1 + \exp\{-(\boldsymbol{c}_j^t)^\top \boldsymbol{v}_i^t\}} \times \boldsymbol{c}_j^t \right] \\ &+ \sum_{(i,k)\notin E^t} \left[\frac{\boldsymbol{c}_k^t}{1 + \exp\{-(\boldsymbol{c}_k^t)^\top \boldsymbol{v}_i^t\}} \right] \\ &+ 2\lambda \boldsymbol{v}_i^t, \end{aligned} \tag{27}$$

Similar to FOPPR, the optimal $\boldsymbol{Z}$ can be obtained by Eq. (22).

6.5 Optimization Algorithm

The loss functions of the four methods have to be minimized over more than one variable. We thus solve each of them by an alternating algorithm [6] which replaces a complex optimization problem with a sequence of easier sub-problems, and then solves the sub-problems alternatingly. In our case, the sub-problems that are not convex, e.g., the sub-problem of FOPPS and SOPPS w.r.t. to each network-specific embedding, can be solved by gradient-based algorithms, e.g., steepest descent or L-BFGS. For those sub-problems that are convex, e.g., the sub-problems of both FOPPR and SOPPR with respect to consensus embeddings, we have obtained the optimal solutions in Eq. (22).

The flowcharts of the alternating algorithms for the four methods are similar, and hence we only present a unified one for all of them. The pseudo-codes of the unified algorithm are presented in Algorithm 1. Algorithm 1 starts with pre-training network-specific embeddings. Pre-training is an important part of an optimization algorithm as it can initialize a model to a

Table 1: Network statistics

Node	Author			Photo			Gene		
Network	Coauthor	Citation	KNN	Tag	Group	Location	Protein	Compound	Disease
#Nodes		6482			4546			8431	
#Edges	19265	120760	421330	169954	74803	13916	28004	137234	176126

point in parameter space that renders the learning process more effective [5]. In our case, the pre-training initializes network-specific embeddings by embedding the corresponding network. The loss functions for the single network embedding have been presented before, i.e., Eq. (4) and Eq. (11) for the first-order proximity preserving and the second-order proximity preserving, respectively. And the derivatives for minimizing them have also been obtained in Eq. (19) and Eq. (27), respectively. For all gradient descent algorithms, the descent rate in each iteration is obtained by backtracking line search [1].

The input r, referred to as negative ratio, is the ratio of the number of pairs of nodes not connected to the number of pairs of nodes connected. The negative ratio is introduced to reduce the computation of the algorithm because the total number of pairs of nodes not connected by edges is the square of the number of nodes, which can be very large in large-scale networks. By referring to the derivatives, the complexity of Algorithm is $O(HL\sum_t |E^t|(1+r))$ where H is the number of outer iterations.

The alternating algorithm is essentially a block-wise coordinate descent algorithm [26] with all the network-specific embeddings and the common embedding or the consensus embedding as block variables. So convergence can be guaranteed based on the general proof of convergence for block-wise coordinate descent. Moreover, we observe that Algorithm 1 converges very fast in terms of the outer iterations in our experiments, which is presented in the evaluation section.

7 Empirical Evaluation

7.1 Datasets

Three real-world datasets are used for the evaluation, and we construct three networks for each dataset presented as follows:

– DBLP dataset [24]: The three networks are author citation network, co-authorship network, and KNN network of authors. In the KNN network, the similarity among authors is quantified by cosine similarity of term frequency of keywords in their papers' abstract. Keywords with overall frequency less than 7 are omitted. The first 1% authors are set as the number k. The papers are selected from popular conferences of four research fields, which are SIGMOD, VLDB, ICDE, EDBT, and PODS for Database, KDD, ICDM, SDM, and PAKDD for Data Mining, ICML, NIPS, AAAI, IJCAI and ECML for Machine Learning, and SIGIR, WSDM, WWW, CIKM, and ECIR for Information Retrieval. Moreover, the publication time is limited from 2000 to 2009. For the selection of authors, authors with papers less than two are omitted.
– Flickr dataset [16]: The three networks are photo networks sampled from CLEF dataset, where the links are established between photos sharing the same tag, group and location, respectively. The three networks are referred to as Photo(Tag) network, Photo(Group) network, and Photo(Location) network in the rest of the paper. Tags with frequency less than 4 and more than 80, and groups with frequency less than 3 and 100, and locations with frequency more than 100 are omitted.
– SLAP dataset [8]: SLAP is a dataset on gene and related objects, such as protein and disease. The three networks are gene networks. Gene interactions usually depend on other objects, such as proteins and diseases. When two genes are related to a particular protein, they have commonly known protein-protein interactions (PPI). Besides proteins and diseases, the third object we employ to establish gene interactions is the chemical compound binding to them. Genes have as least one interaction incurred by disease or chemical compound are selected in our experiments. Similarly, the three networks are referred to as Gene(PPI) network, Gene(DDI) network, and Gene(CCI) network.

Network statistics are summarized in Table 1.

7.2 Experiment Settings

Five recent network embedding models mentioned in the related work are used as baselines, which are DeepWalk [21], LINE [25], TADW [38], node2vec [13], and EOE [34]. However, because TADW is text-associated DeepWalk, and EOE is for embedding coupled heterogeneous networks, we only use them as baselines on the DBLP dataset where the keywords of papers can be associated with the co-authorship network or the citation network to fit these two models.

For the implementation of Algorithm 1, the dimension of embeddings is set as 128, which is commonly used in all the baselines, the negative ratio is set as 5 as used in LINE, all the regularization coefficients are set as 1.0, and common settings are used in backtracking line search. Our codes are written in Java and run on an Intel Genuine Intel(R) CPU @2.60GHZ 2.60GHZ server with 64 GB RAM.

In all the evaluations below, network-specific embeddings learned by FOPPS and SOPPS are evaluated after being combined with corresponding common embeddings because neither the network-specific embeddings nor the common embeddings are complete embeddings for a particular network. Correspondingly, all the network-specific embeddings learned by FOPPR and SOPPR and the consensus embeddings are complete embeddings, and they are not combined for the evaluation.

7.3 Network Visualization

This section presents visual evaluation of embeddings in a two-dimensional space. If the embeddings preserve the network structure well, the visualization should display a layout similar to that of the original network. We employ the author networks to illustrate this point because these networks are well structured in a way that researchers of the same field tend to have co-authorships and citations. The dimension of embeddings is 128, we thus employ t-SNE tool [15] to project them into 2-dimension vectors. The visualization results are presented in Fig. 2. Only visualizations of embeddings learned from the co-authorship network are presented because visualizations of embeddings from the other two networks show similar patterns, and LINE(1st) has been omitted because it underperforms LINE(2nd).

From Fig. 2(1) to Fig. 2(6) which are visualizations obtained by models for embedding a single network, we see that the researchers of different fields are largely mixed up. This is because the selected four fields, i.e., DB, DM, ML, and IR, are closely related, and hence they are many cross-field co-authorships. But EOE and the proposed models for jointly embedding multiple networks perform better by utilizing information of keywords of their papers which are usually domain-specific. Domain-specific keywords can be demonstrated by Fig. 2(12) where the visualization of pre-trained embeddings learned from the DBLP KNN network shows four clearly distinguishable clusters. It is not conclusive which mechanism to combine the co-authorship network with keywords is better because there is no clearly visual advantage of one over another. But the effectiveness of the proposed mechanism can be clearly seen by comparing pre-trained embeddings in Fig. 2(5) or Fig. 2(6) with final embeddings in Fig. 2(8) and Fig. 2(10) or Fig. 2(9) and Fig. 2(11).

TADW in Fig. 2(13) performs even better than the pre-trained embeddings learned from the DBLP KNN network because it directly concatenates embeddings learned from the co-authorship network and the embeddings learned from keywords. In this case, the dimension of embeddings of TADW is 128×2, which seems to have unfair advantages. The concatenated embeddings of each baseline can also have similar performance, but are not presented due to limited space. It is worthy of noting that the consensus embeddings learned by FOPPR and SOPPR in Fig. 2(13) and Fig. 2(14) are comparable to TADW even though the dimension is 128. This is because the consensus embeddings can effectively summarize all the network-specific embeddings. Although the consensus embeddings appear to be the average of network-specific embeddings as indicated in Eq. (22), they are actually optimized by jointly embedding multiple networks. To demonstrate this point, the average of three network-specific embeddings learned by node2vec are visualized in Fig. 2(16), and the average of other baselines is omit because it shows similar results. We see that the visualization seems to be randomized. This is because the network-specific embeddings which are learned separately are actually not comparable.

7.4 Link Prediction

Link prediction refers to inferring new interactions between network nodes, and commonly used inference methods are based on similarities between nodes as nodes sharing similar characteristics are more likely to interact [14]. Similarity metrics for link prediction, such Common Neighbors and Jaccard's Coefficient, are not used as baselines because they significantly underperform node2vec as indicated in its paper [13]. Inner product of two node embeddings normalized by sigmoid function is employed as the similarity measurement. Except for the KNN network of researchers, all the other networks are used in the link prediction task. For the DBLP networks, we predict future co-authorships and citations that occur from 2010 to 2013 while for all the other networks, we predict missing links that are not used as training information. For all networks, the same number of non-existing links are randomly sampled for the evaluation purpose. One thing to clarify for the proposed models is that , if there is no specification, network-specific embeddings are employed to

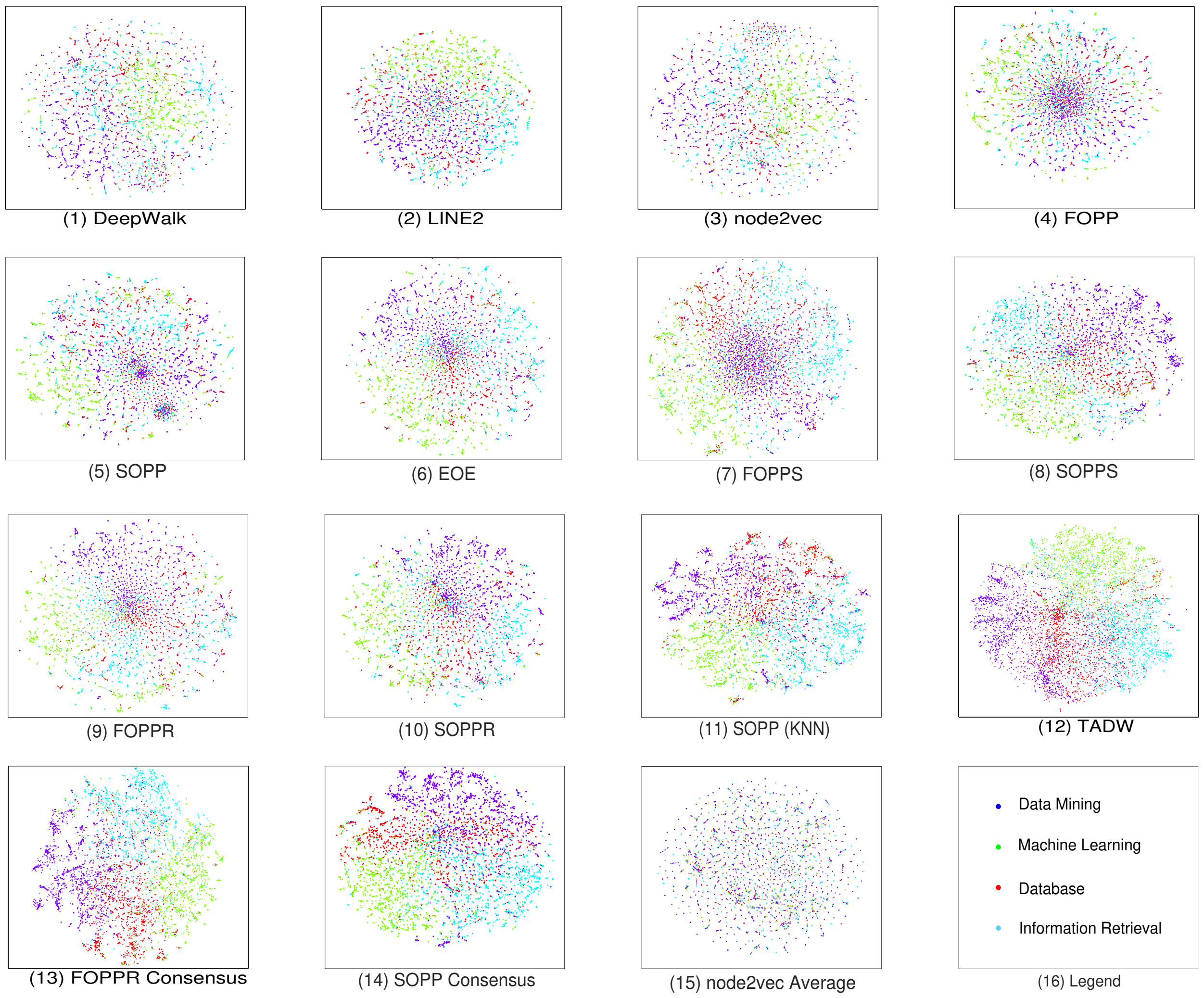

Fig. 2: Visualizations of embeddings corresponding to the DBLP co-authorship network by different models, where different colors denote different research fields of researchers as shown in (16).

perform each link prediction task, e.g., for the DBLP co-authorship prediction, network-specific embeddings corresponding to the co-authorship network are used.

The AUC scores for tasks on the co-authorship network and the citation network are presented in Table 2. It shows that the proposed models perform better than all the baselines. The proposed models perform better than baselines for embedding a single network because each task can utilize knowledge learned from related tasks, i.e., embeddings learned from the co-authorship network can be improved by knowledge learned from the citation network and the KNN network. The benefits of utilizing more related information can also be seen in the superior performance of EOE to other baselines. But EOE underperforms the proposed models because it can only utilize two sources of information.

It is worthy of noting that the way of utilizing information from other networks is important. To demonstrate this point, we combine all the networks into one network, and then apply single-task baselines to the combined network. We present the corresponding AUC scores in Table 3. It shows that most methods underperform themselves when only a single network is utilized in most times. This is because the combined network enforces all the types of interactions to be indifferent. Particularly for this case, all the researchers may be very similar when the similarity is largely measured by the keywords of their papers because the selected four fields are closely related. However, it is not appropriate for link prediction. This also explains why TADW which directly concatenates embeddings learned from keywords into node embeddings also underperforms DeepWalk. Hence, in the following link prediction tasks, we omit

Table 2: AUC scores (100%) for link prediction tasks on the co-authorship network and the citation network.

AUC	DeepWalk	LINE(1st)	LINE(2nd)	node2vec	TADW	EOE
Co-authorship	74.07	62. 39	75.36	71.07	56.25	76.69
Citation	83.97	80.32	84.88	84.66	58.62	85.36
AUC	FOPPS	FOPPR	FOPPR Consensus	SOPPS	**SOPPR**	**SOPPR Consensus**
Co-authorship	78.29	77.62	78.92	80.95	**81.29**	79.50
Citation	86.54	85.80	87.84	89.45	88.10	**90.90**

Table 3: AUC scores (100%) for link prediction tasks on the co-authorship network and the citation network when the baselines are applied to the network combined from all the three networks.

AUC	DeepWalk	LINE(1st)	LINE(2nd)	node2vec
Co-authorship	75.37	70.06	72.01	75.32
Citation	79.31	76.46	76.63	79.55

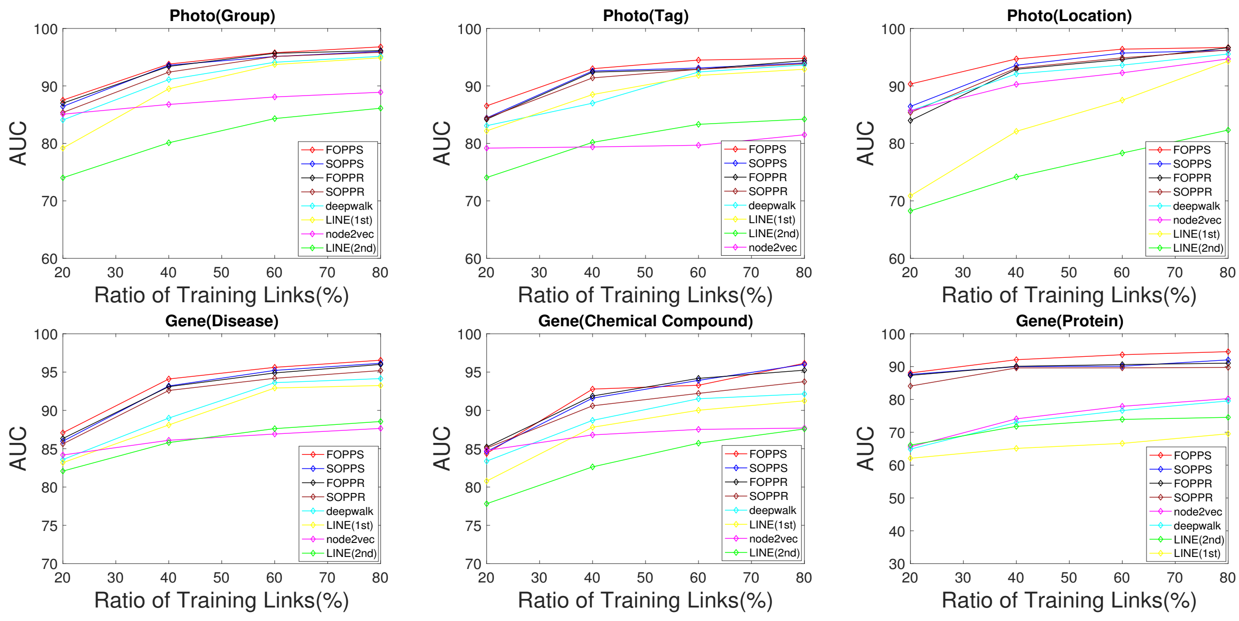

Fig. 3: AUC plots on link prediction tasks, where methods are placed according to their ranking of performance in the legend.

the performance obtained by the single-task baselines on the corresponding combined network.

One interesting observation is that the consensus embeddings learned by SOPPR perform the best in the co-authorship prediction. Recall that the consensus embeddings summarize all the network-specific embeddings. Then the observation suggests that overall knowledge performs better than network-specific knowledge. We notice that the co-authorship network, the citation network, and the KNN network of researchers are closely related in the sense that links in all the three networks are established between researchers sharing common research interests. In this case, the summation of all network-specific knowledge may be superior to each of its components, which is also suggested in the visualization. The common embeddings learned by FOPPS and SOPPS alone perform the worst because they only contain partial information, and are omit here and in the rest of link prediction tasks.

For the photo networks and the gene networks, we conduct 4 runs of experiments in which different ratios of links are used as training links and the remaining ones are used as test links. The AUC scores are plotted in Fig. 3. The methods in the legend are placed according to their rankings of performance. Similarly, the proposed models perform better than all the baselines. Moreover, the advantage is more visible when less links are used in the training phase, which suggests complementary information from other networks is especially important when linkage information of a particular network is limited.

Table 4: Micro-F1 and Macro-F1 scores (100%) for multi-label classification.

	Model	LINE(1st)		LINE(2nd)		DeepWalk		node2vec		TADW	
	Feature	short	long	short	long	short	long	short	long	Coauthor	Citation
DBLP	Micro-F1	66.6	65.8	72.1	69.0	73.7	72.0	72.0	69.2	81.2	82.0
	Macro-F1	63.6	62.8	67.9	67.1	68.7	67.2	67.1	64.5	73.1	73.5
Flickr	Micro-F1	61.6	59.9	64.8	63.2	65.4	64.9	65.5	65.2	–	–
	Macro-F1	61.1	59.6	64.2	62.7	65.1	64.7	65.1	64.7	–	–

	Model	EOE		FOPPS		**FOPPR**		SOPPS		**SOPPR**	
	Feature	Coauthor	Citation	short	long	short	long	short	long	short	long
DBLP	Micro-F1	79.0	80.4	84.2	84.3	84.5	**86.2**	85.4	85.5	85.8	86.0
	Macro-F1	72.2	72.9	75.2	75.1	75.8	**78.4**	77.7	77.9	78.1	78.3
Flickr	Micro-F1	–	–	66.1	65.1	67.2	67.1	66.8	67.0	68.0	**68.3**
	Macro-F1	–	–	65.9	64.9	67.1	66.9	66.5	66.9	67.3	**67.5**

Unlike in Table 2 where the consensus embeddings obtain the best performance, network-specific embeddings learned by FOPPS obtain the best performance. Moreover, network-specific embeddings learned by other models also perform better than the consensus embeddings. Hence, the performance of consensus embeddings are not included in Fig. 3. Network-specific embeddings perform better than the consensus embeddings may be that the three photo networks and the gene networks are not that closely related like the three author networks. More specifically, the links in different photo networks or genes networks are established due to different properties, e.g., links in Photo(Tag) network are established between photos sharing similar descriptions such as "scenery" while links in Photo(Location) network are established between photos sharing the same location. It is worthy of noting that a location is far beyond the descriptions for a photo taken in the location. Hence, the links are essentially different among the three photo networks as well as gene networks.

Models with common embeddings, i.e., FOPPS and SOPPS, outperform modes with consensus embeddings, i.e., FOPPR and SOPPR, in most times. This may be because the common characteristics are more explicitly combined with network-specific embeddings in each task in the former type of models than in the latter type.

7.5 Multi-label Classification

This section evaluates embeddings as features used in multi-label classification where more than one label are assigned to each data point. The four research fields are used as the labels for the DBLP dataset. For the Flickr dataset, the 99 labels indicated in the dataset are all employed in the evaluation. We employ binary-relevance SVM with polynomial kernel implemented in Meka [23] as the classifier, and use 5-fold cross validation as the evaluation method. There are two types of settings for the embeddings used as features. The first setting is for all the models except for TADW and EOE. All the network-specific embeddings are directly concatenated as features. Besides network-specific embeddings, the proposed models have common embeddings and consensus embeddings. To demonstrate the effectiveness of these information-sharing embeddings, we prepare two set of features, i.e., features without common embeddings or consensus embeddings and features with common embeddings or consensus embeddings, which are referred to as short features and long features, respectively. The short features are concatenated to common embeddings or consensus embeddings to form the long features. To make the comparison fair in the sense that the dimension of features is the same, we also prepare short features and long features for baselines where long features are short features plus the average of network-specific embeddings.

The second setting is for TADW and EOE. For these two models, we don't do any manipulation on the embeddings because they can utilize two sources of information at the same time. But the keywords can be combined to both the co-authorship network and the citation network, the embeddings learned from the co-authorship network are used as one set of features and the embeddings learned from the citation network are used as another set of features.

The results of Micro-F1 and Macro-F1 scores are presented in Table 4. It shows the proposed models perform better than all the baselines regardless of whether or not the common embeddings or the consensus embeddings are used, especially on the DBLP dataset. For the baselines except for TADW and EOE, all the long features underperform the short features. This suggests that the direct average of network-specific embeddings is not appropriate. However, the long feature of the proposed models performs better than the short feature on the DBLP dataset and almost ties with the short feature on the Flickr dataset. Hence, the consensus embedding is an effective way to summarize network-specific embeddings, and thus may provide additional information for the task. The effectiveness of the consensus embedding in summarizing network-specific embeddings has been demonstrated in visualizations illustrated in Fig. 2(14) and Fig. 2(15).

The proposed models outperform TADW and EOE because they can utilize one more source of information than TADW and EOE. Hence, we can image that when the number of networks is more than three, the advantage of the proposed models may be even larger.

With respect to the comparison among the proposed models, it shows SOPP-based models perform better than FOPP-based models, and models with consensus embeddings perform better than models with common embeddings except that FOPPR slightly outperforms SOPPR on the DBLP data when long feature is used.

7.6 Summarization of Comparison

We summarize the comparison for the selection of an appropriate one for a particular application. It is not easy to obtain the advantages or disadvantages of different models from theoretical aspect. We thus perform the comparison mainly from empirical aspect.

With respect to the mechanism for enforcing the information-sharing embedding, FOPPS and SOPPS belong to one group while FOPPR and SOPPR belong to another group. The former group enforces the information-sharing embedding as a common embedding shared by all network-specific embeddings while the latter group enforces it as a consensus embedding on which all network-specific embeddings agree. Experiments show that the former group consistently outperforms the latter group for the link prediction tasks except that SOPPS is comparable to SOPPR for the DBLP co-authorship prediction while the former group consistently underperforms the latter group for the multi-label classification tasks. It is worthy of noting that link prediction is network-specific while multi-label classification is not. Hence, the former group performs better on link prediction because the common embedding is a more effective way for sharing information across multiple networks. And the latter group performs better on multi-label classification because the consensus embedding is a more effective way to fuse information.

With respect to the mechanism for embedding the network structure, FOPPS and FOPPR belong to one group while SOPPS and SOPPR belong to another group. The former group embeds the network structure by preserving first-order proximity while the latter group preserves second-order proximity. Experiments show that for link prediction on DBLP, the latter group

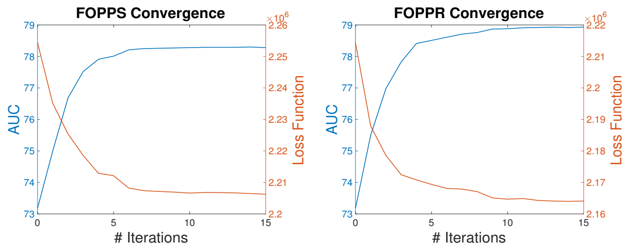

Fig. 4: Convergence analysis

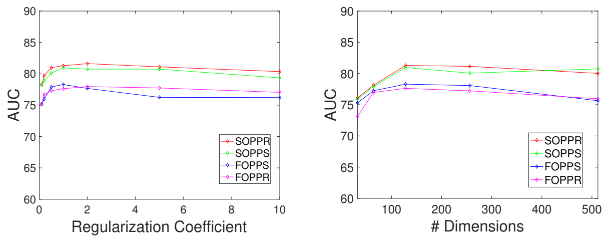

Fig. 5: Parameter analysis where network-specific embeddings are used in all the tasks.

outperforms the former group, but for link prediction on the Flickr networks and the Gene networks, the former group outperforms the latter group in most times. This pattern can be also seen in the comparison of LINE(1st) and LINE(2nd), which embeds the network structure via preserving the first-order proximity and the second-order proximity, respectively. Hence, there is no conclusion about which mechanism for embedding the network structure is better for link prediction.

But for multi-label classification, the experiments show the latter group consistently outperforms the former group except that SOPPR is comparable to FOPPR on the DBLP multi-label classification suggested by Table 3. Unfortunately, we have not obtained valid reasons why these two mechanisms have different performance on different tasks.

To sum up, for network-specific applications, such as link prediction, we suggest using FOPPS and SOPPS. For those applications that are not specific to a particular network, we suggest using SOPPR.

7.7 Convergence Analysis

This section studies the convergence of the proposed alternating algorithm as indicated in Algorithm 1. Specifically, we study the performance of the algorithm on applications with respect to the number of outer iterations. We only present the performance on the link prediction task for the DBLP co-authorship network in Fig. 4 obtained by FOPPS and FOPPR because other experiments show similar results. The performance of

FOPPS is obtained by network-specific embeddings while the performance of FOPPR is obtained by the consensus embeddings because those embeddings achieve the best performance of the models. It shows that the algorithm converges very fast and can usually converge to stable performance after about 10 iterations.

7.8 Parameter Sensitivity

This section evaluates how sensitive the proposed models are to the regularization coefficients and the dimension of embeddings. The performance on the DBLP coauthorship prediction w.r.t. regularization coefficients and the dimension of embeddings is presented in Fig. 5. The studied coefficients range from 0.01 to 10, and the dimension ranges from 32 to 512. The figure suggests that the regularization coefficients should better within the range [0.05, 2], and the dimension should better within the range [64, 256];

8 Conclusion and Future Work

This paper proposes for the first time to jointly embed multiple networks which share the same set of nodes. To share the characteristics of node exhibited in one network to another network, we propose to enforce an information-sharing embedding. The information-sharing embedding is employed as a common embedding in FOPPS and SOPPS and as a consensus embedding in FOPPR and SOPPR. Through comprehensive experiments, we demonstrate the proposed models outperform five recent network embedding models in applications including visualization, link prediction, and multi-label classification. In the future, we plan to enable the proposed models to perform online learning so as to handle evolving networks.

Acknowledgements The work described in this paper was partially supported by National Key R&D Program of China - 2018 YFB1004801, RGC General Research Fund under Grant PolyU 152199/17E, the funding for Project of Strategic Importance provided by The Hong Kong Polytechnic University (Project Code: 1-ZE26), NSF through grants IIS-1526499, IIS-1763325, and CNS-1626432, and NSFC 61672313.

References

1. Armijo L (1966) Minimization of functions having lipschitz continuous first partial derivatives. Pacific Journal of mathematics 16(1):1–3
2. Bakker B, Heskes T (2003) Task clustering and gating for bayesian multitask learning. Journal of Machine Learning Research 4(May):83–99
3. Bakshy E, Rosenn I, Marlow C, Adamic L (2012) The role of social networks in information diffusion. In: Proceedings of the 21st international conference on World Wide Web, ACM, pp 519–528
4. Belkin M, Niyogi P (2001) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: NIPS, vol 14, pp 585–591
5. Bengio Y, Lamblin P, Popovici D, Larochelle H, et al (2007) Greedy layer-wise training of deep networks. Advances in neural information processing systems 19:153
6. Bezdek JC, Hathaway RJ (2002) Some notes on alternating optimization. In: AFSS International Conference on Fuzzy Systems, Springer, pp 288–300
7. Bhagat S, Cormode G, Muthukrishnan S (2011) Node classification in social networks. In: Social network data analytics, Springer, pp 115–148
8. Chen B, Ding Y, Wild DJ (2012) Assessing drug target association using semantic linked data. PLoS Comput Biol 8(7):e1002,574
9. Collobert R, Weston J (2008) A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning, ACM, pp 160–167
10. Cox TF, Cox MA (2000) Multidimensional scaling. CRC press
11. Dong Y, Chawla NV, Swami A (2017) metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM
12. Evgeniou T, Pontil M (2004) Regularized multi–task learning. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 109–117
13. Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
14. Liben-Nowell D, Kleinberg J (2007) The link-prediction problem for social networks. journal of the Association for Information Science and Technology 58(7):1019–1031
15. Maaten Lvd, Hinton G (2008) Visualizing data using t-sne. Journal of Machine Learning Research 9(Nov):2579–2605
16. McAuley J, Leskovec J (2012) Image labeling on a network: using social-network metadata for image classification. In: European Conference on Computer Vision, Springer, pp 828–841

17. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781
18. Ni J, Tong H, Fan W, Zhang X (2015) Flexible and robust multi-network clustering. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp 835–844
19. Ou M, Cui P, Pei J, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: KDD, pp 1105–1114
20. Parameswaran S, Weinberger KQ (2010) Large margin multi-task metric learning. In: Advances in neural information processing systems, pp 1867–1875
21. Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 701–710
22. Philip SY, Zhang J (2015) Mcd: Mutual clustering across multiple social networks. In: Big Data (BigData Congress), 2015 IEEE International Congress on, IEEE, pp 762–771
23. Read J, Reutemann P, Pfahringer B, Holmes G (2016) Meka: a multi-label/multi-target extension to weka. Journal of Machine Learning Research 17(21):1–5
24. Tang J, Zhang J, Yao L, Li J, Zhang L, Su Z (2008) Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM
25. Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015) Line: Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, pp 1067–1077
26. Tseng P (2001) Convergence of a block coordinate descent method for nondifferentiable minimization. Journal of optimization theory and applications 109(3)
27. Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017) Community preserving network embedding. In: AAAI, pp 203–209
28. Wei X, Xie S, Yu PS (2015) Efficient partial order preserving unsupervised feature selection on networks. In: Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015, pp 82–90
29. Wei X, Cao B, Shao W, Lu C, Yu PS (2016) Community detection with partially observable links and node attributes. In: 2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016, pp 773–782
30. Wei X, Xu L, Cao B, Yu PS (2017) Cross view link prediction by learning noise-resilient representation consensus. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017, pp 1611–1619
31. Xu L, Wei X, Cao J, Philip SY (2017) Disentangled link prediction for signed social networks via disentangled representation learning. In: 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, pp 676–685
32. Xu L, Wei X, Cao J, Philip SY (2017) Multitask network embedding. In: Data Science and Advanced Analytics (DSAA), 2017 IEEE International Conference on, IEEE, pp 571–580
33. Xu L, Wei X, Cao J, Yu PS (2017) Embedding identity and interest for social networks. In: Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017, pp 859–860
34. Xu L, Wei X, Cao J, Yu PS (2017) Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, pp 741–749
35. Xu L, Wei X, Cao J, Yu PS (2017) Multiple social role embedding. In: Proceedings of the international conference on data science and advanced analytics, IEEE
36. Xu L, Wei X, Cao J, Philip SY (2018) Interaction content aware network embedding via coembedding of nodes and edges. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, pp 183–195
37. Xu L, Wei X, Cao J, Yu PS (2018) On exploring semantic meanings of links for embedding social networks. In: Proceedings of the 2018 World Wide Web Conference on World Wide Web, International World Wide Web Conferences Steering Committee, pp 479–488
38. Yang C, Liu Z, Zhao D, Sun M, Chang EY (2015) Network representation learning with rich text information. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, pp 2111–2117
39. Zhang J, Philip SY (2015) Integrated anchor and social link predictions across social networks. In: IJCAI, pp 2125–2132
40. Zhang J, Yu PS, Zhou ZH (2014) Meta-path based multi-network collective link prediction. In: Pro-

ceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 1286–1295