The following publication L. Xu, X. Wei, J. Cao and P. S. Yu, "Multi-task Network Embedding," 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Tokyo, Japan, 2017, pp. 571-580 is available at https://doi.org/10.1109/DSAA.2017.19.

# Multi-task Network Embedding

Linchuan Xu*, Xiaokai Wei†, Jiannong Cao* and Philip S. Yu‡
*The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
Email: 13901626r@connect.polyu.hk, csjcao@comp.polyu.edu.hk
†Facebook Inc., 1 Hacker Way, Menlo Park, CA, USA
Email: weixiaokai@gmail.com
‡University of Illinois at Chicago, Chicago, Illinois, USA
Email: psyu@uic.edu

***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 (MTNE) to jointly learn multiple network-specific embeddings for each node via enforcing an extra information-sharing embedding. Moreover, we instantiate the idea in two models that are different in the mechanism for enforcing the information-sharing embedding. The first model 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 model enforces the information-sharing embedding as a consensus embedding on which all network-specific embeddings agree. We demonstrate through comprehensive experiments on three real-world datasets that the proposed models outperform state-of-the-art network embedding models in applications including visualization, link prediction, and multi-label classification.**

## I. 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 [15] [27], network visualization [16], information diffusion [3], and node classification [8]. 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 [19] [22] [14] [30] [29] [28] [26]. However, most existing 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 joint multiple online social networks, such as Facebook, Twitter, and LinkedIn.

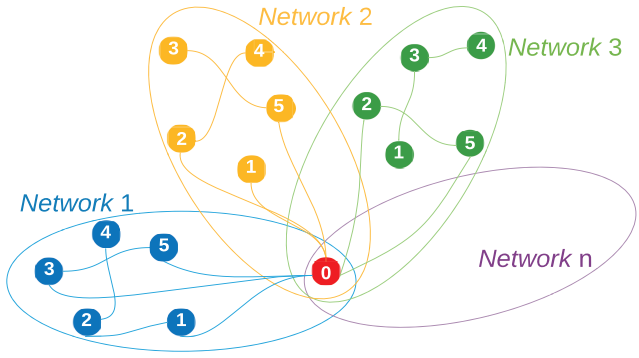


Fig. 1: A toy example of a node participating in multiple undirected networks, which is drawn in the form of the ego network from the perspective of node 0.

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, such as 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 scenario, a task corresponds to 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 MTNE idea is instantiated in two

models, which are different in the mechanism for enforcing the information-sharing embedding.

The first model enforces the information-sharing embedding as a common embedding, which is similar to the concept of the common metric in multi-task metric learning [18]. 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 [13], multi-task deep learning [11], and multi-task distance metric learning [18]. 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. This model is then further referred to as MTNE-C where C is for combination, and the mechanism for the combination is introduced in the following sections.

The second model 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. This model is thus referred to as MTNE-R in the rest of paper where R is for regularization.

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 in MTNE-C. Moreover, this work proposes another novel way of capturing and utilizing the commonalties shared by multiple networks, which is instantiated in MTNE-R.
- We conduct comprehensive experiments on three real-world datasets to demonstrate that the proposed models outperform the start-of-the-art 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.

## II. Related Work

The first category of related work is network embedding which learns node representations by preserving the network structre. Previously, network embedding [12] [24] [21] [4] has been proposed to reduce the dimensionality of features because high-dimensional features make it inefficient or even infeasible to train data mining models. In this scenario, they have to construct a network between isolated data points before performing embedding, where typical networks are nearest neighbor networks.

Recently, network embedding methods under the natural network scenario have been proposed, and five representatives are DeepWalk [19], LINE [22], TADW [31], node2vec [14], and EOE [30] in chronological order. DeepWalk [19] learns node embeddings by presenting sequences of nodes obtained by random walks to be close in the embedding space. TADW [31] learns embeddings in a way similar to DeepWalk but enriches the embeddings with user-generated text. LINE [22] presents pairs of nodes with first-order links or second-order links to be close in the embedding space. The key distinction of node2vec [14] model is a flexible definition of neighborhood so as to explore diverse neighbors of nodes. Although EOE [30] jointly embeds two networks, the two networks do not share the same set of nodes, and moreover, they are interconnected. In our case, multiple networks share the same set of nodes so that there are no connections between any two networks.

The second category of related work is multi-task learning [9] [2] [5] [11] which jointly learns multiple task-specific knowledge via enforcing a pool of shared knowledge, such as a shared parameterization or representation. The proposed MTNE-C learns a shared embedding for each node in a way similar to the shared knowledge learning in multi-task SVM [13] and multi-task metric learning [18]. However, instead of learning a shared embedding, the proposed MTNE-R learns a consensus embedding for sharing information across multiple tasks.

## III. Multi-task Metric Learning

In this section, we briefly review the multi-task distance metric learning [18] for the introduction of the proposed MTNE-C. 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 measurement 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 multi-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:

$$d_t(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T (\boldsymbol{M}_t + \boldsymbol{M}_0)(\boldsymbol{x}_i - \boldsymbol{x}_j)}, \quad (1)$$

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 [18]. In the following sections, we show how this elegant formulation can be adapted to multi-task network embedding.

## IV. Preliminaries for MTNE

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 shared by all networks and $E^t$ representing a set of network-specific weighted or unweighted, directed or undirected edges.

As an embedding method, MTNE embeds the network structure in an Euclidean space. The embedding of network structure is achieved by presenting nodes connected by edges to be close in the embedding space because nodes connected by edges are close in the network representation. The closeness in the embedding space is defined as follows:

**DEFINITION 2.** The **closeness** between two nodes for network $t$ in the embedding space is defined as the probability that there exists an edge between them, where the probability is computed on their embeddings as follows:

$$p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t) = \frac{1}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}}, \quad (2)$$

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 larger probabilities indicate larger inner products of two vectors, and the inner product is a measurement of closeness of two points in Euclidean space. Moreover, the closeness defined in this way is to facilitate the formulation of optimization objective presented below.

To preserve the network structure in the embedding space, MTNE presents nodes not connected by edges to be apart in the embedding space as well. The non-linkage information is also an important part of the network structure because it indicates certain nodes are not likely to interact. Hence, the proposed ***network structure preserving*** mechanism is summarized as follows:

- Nodes connected by edges are presented to be close in the embedding space
- Nodes not connected are presented to be apart in the embedding space

Since the closeness is defined as probabilities of the existence of edges, the proposed network structure preserving mechanism applied to a single network can be formulated into an optimization objective according to the maximum likelihood estimation as follows:

$$\max_{\boldsymbol{V}^t} \prod_{(i,j) \in E^t, (h,k) \notin E^t} p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t)(1 - p(\boldsymbol{v}_h^t, \boldsymbol{v}_k^t)), \quad (3)$$

which maximizes the probabilities of edges among the nodes in network $G^t(V, E^t)$. 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 $G^t$.

The multiplication maximization problem is usually transferred to an equivalent minimization problem by assigning a negative sign and taking natural logarithm, which is denoted as follows:

$$\min_{\boldsymbol{V}^t} l(\boldsymbol{V}^t), \quad (4)$$

where

$$l(\boldsymbol{V}^t) = -\sum_{(i,j) \in E^t} w_{ij}^t \log p(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t) - \sum_{(h,k) \notin E^t} \log(1 - p(\boldsymbol{v}_h^t, \boldsymbol{v}_k^t)), \quad (5)$$

where $w_{ij}^t \in \mathbb{R}$ is the weight of edge $(i, j)$ to reflect to relationship strength, and does not violate the original objective as indicated in Eq. (3).

## V. Multi-task Network Embedding via Enforcing a Common Embedding

This section extends the proposed single-task network embedding to multi-task network embedding via enforcing an information-sharing embedding. Specifically, the information-sharing embedding is employed a common embedding, which is the similar to the concept of the common metric in multi-task metric learning. The common embedding is defined as follows:

**DEFINITION 3.** 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 closeness for network $t$ in the multi-task setting is quantified as follows:

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

In this way, the common embedding and a network-specific embedding are combined to be a complete embedding for the closeness measurement. As a result, the model is named MTNE-C where "C" is for combination. 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 closeness can be more comprehensive as it utilizes information from other networks.

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:

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

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

## VI. Multi-task Network Embedding via Enforcing a Consensus Embedding

This section presents an alternative way to incorporate the information-sharing embedding other than as a common embedding. Specifically, the information-sharing embedding is employed as a consensus embedding, which is defined as follows:

**DEFINITION 4.** 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:

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

which minimizes the difference between network-specific embeddings and the consensus embedding. Hence, the model with the consensus embedding is named MTNE-R where "R" is for regularization.

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

$$l^R = \sum_t l(\boldsymbol{V}^t) + \gamma \sum_{t=1} ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 + \lambda \sum_t ||\boldsymbol{V}^t||_F^2 + \beta ||\boldsymbol{Z}||_F^2, \tag{9}$$

where $l(\boldsymbol{V}^t)$ is quantified in Eq. (5), and $\gamma \in \mathbb{R}$. 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.

## VII. The Optimization

### A. Optimization for MTNE-C

Firstly, we compute the derivative for minimizing $l^C$. The derivative w.r.t $\boldsymbol{v}_i^t$ is computed as follows: $\frac{\partial l^C}{\partial \boldsymbol{v}_i^t} =$

$$\begin{aligned} & - \sum_{(i,j)\in E^t} \left[ \frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}} \times (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0) \right] \\ & + \sum_{(i,k)\notin E^t} \left[ \frac{(\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)\}} \right] \\ & + 2\lambda \boldsymbol{v}_i^t, \end{aligned} \tag{10}$$

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

$$\begin{aligned} & \sum_d \Bigg[ - \sum_{(i,j)\in E^t} \left[ \frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0)\}} \times (\boldsymbol{v}_j^t + \boldsymbol{v}_j^0) \right] \\ & + \sum_{(i,k)\notin E^t} \left[ \frac{(\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)}{1 + \exp\{-(\boldsymbol{v}_i^t + \boldsymbol{v}_i^0)^\top (\boldsymbol{v}_k^t + \boldsymbol{v}_k^0)\}} \right] \Bigg] \\ & + 2\beta \boldsymbol{v}_i^0, \end{aligned} \tag{11}$$

### B. Optimization for MTNE-R

To derive the derivative of $l^R$ with respect to a particular network-specific embedding, $l^R$ reduces to the following function:

$$l^R = l(\boldsymbol{V}^t) + \beta ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 \tag{12}$$

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

$$\frac{\partial l^R}{\partial \boldsymbol{v}_i^t} = \frac{\partial l(\boldsymbol{V}^t)}{\partial \boldsymbol{v}_i^t} + 2\gamma(\boldsymbol{v}_i^t - \boldsymbol{z}_i), \tag{13}$$

where

$$\begin{aligned} \frac{\partial l(\boldsymbol{V}^t)}{\partial \boldsymbol{v}_i^t} = & - \sum_{(i,j)\in E^t} \left[ \frac{w_{ij}^t \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_j^t\}} \times \boldsymbol{v}_j^t \right] \\ & + \sum_{(i,k)\notin E^t} \left[ \frac{\boldsymbol{v}_k^t}{1 + \exp\{-(\boldsymbol{v}_i^t)^\top \boldsymbol{v}_k^t\}} \right] \\ & + 2\lambda \boldsymbol{v}_i^t, \end{aligned} \tag{14}$$

To compute the derivative with respect to consensus embedding $\boldsymbol{Z}$, the minimization problem reduces to the following one:

$$\min_{\boldsymbol{Z}} \gamma \sum_t ||\boldsymbol{Z} - \boldsymbol{V}^t||_F^2 + \beta ||\boldsymbol{Z}||_F^2, \tag{15}$$

**Algorithm 1:** The alternating optimization algorithm

**Input :** $G = \{G^1, ..., G^t, ..., G^T\}$, L, negative ratio, $\lambda$, $\beta$, and $\gamma$
**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 $\boldsymbol{V}^1, ..., \boldsymbol{V}^t, ..., \boldsymbol{V}^T$;
**while** *(not converge)* **do**
  Solve $\boldsymbol{V}^0$ with gradient descent for MTNE-C, or find the optimal $\boldsymbol{Z}$ with the Eq. (17) for MTNE-R with all network-specific embeddings fixed;
  **for** $t \leftarrow$ *1 to* $T$ **do**
    Find the optimal $\boldsymbol{V}^t$ with gradient descent with all the other variables fixed;
**return** $\boldsymbol{V}^1, ..., \boldsymbol{V}^t, ..., \boldsymbol{V}^T$, and $\boldsymbol{V}^0$ or $\boldsymbol{Z}$

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

$$\gamma \sum_t 2(\boldsymbol{Z} - \boldsymbol{V}^t) + 2\beta \boldsymbol{Z} \tag{16}$$

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

$$\boldsymbol{Z} = \frac{\gamma \sum_t \boldsymbol{V}^t}{\gamma T + \beta} \tag{17}$$

### C. Optimization Algorithm

Both $l^C$ and $l^R$ have to be minimized over more than one variables including network-specific embeddings $\boldsymbol{V}^t$ and common embedding $\boldsymbol{V}^0$ or consensus embedding $\boldsymbol{Z}$. We thus solve them by an alternating algorithm [7] 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 of MTNE-C with respect to $\boldsymbol{V}^t$ and $\boldsymbol{V}^0$ can be solved by gradient-based algorithms, e.g., steepest descent or L-BFGS. The sub-problems of MTNE-R with respect to $\boldsymbol{V}^t$ can similarly solved like those of MTNE-C while the sub-problem with respect to $\boldsymbol{Z}$ is solved in Eq. (17). For all gradient descent algorithms, the descent rate in each iteration is obtained by backtracking line search [1].

The flowchart of the alternating algorithm for METN-C and MTNE-R is similar, and hence we only present a unified one for both of them. The pseudo-codes of the 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 point in parameter space that renders the learning process more effective [6]. In our case, pre-training initializes network-specific embeddings by performing single-network embedding, which can also be achieved by gradient descent on the network-specific loss function Eq. (5) and the derivative is computed in Eq. (14).

The input 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.

The alternating algorithm is essentially a block-wise coordinate descent algorithm [25] 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.

## VIII. Empirical Evaluation

### A. Datasets

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

- DBLP dataset [23]: The three networks are co-authorship network, author citation 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 [17]: 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 [10]: 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 interaction (PPI). Besides proteins and diseases, the third object we employ to establish gene interactions is the common 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.

| 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 |

TABLE I: Network statistics

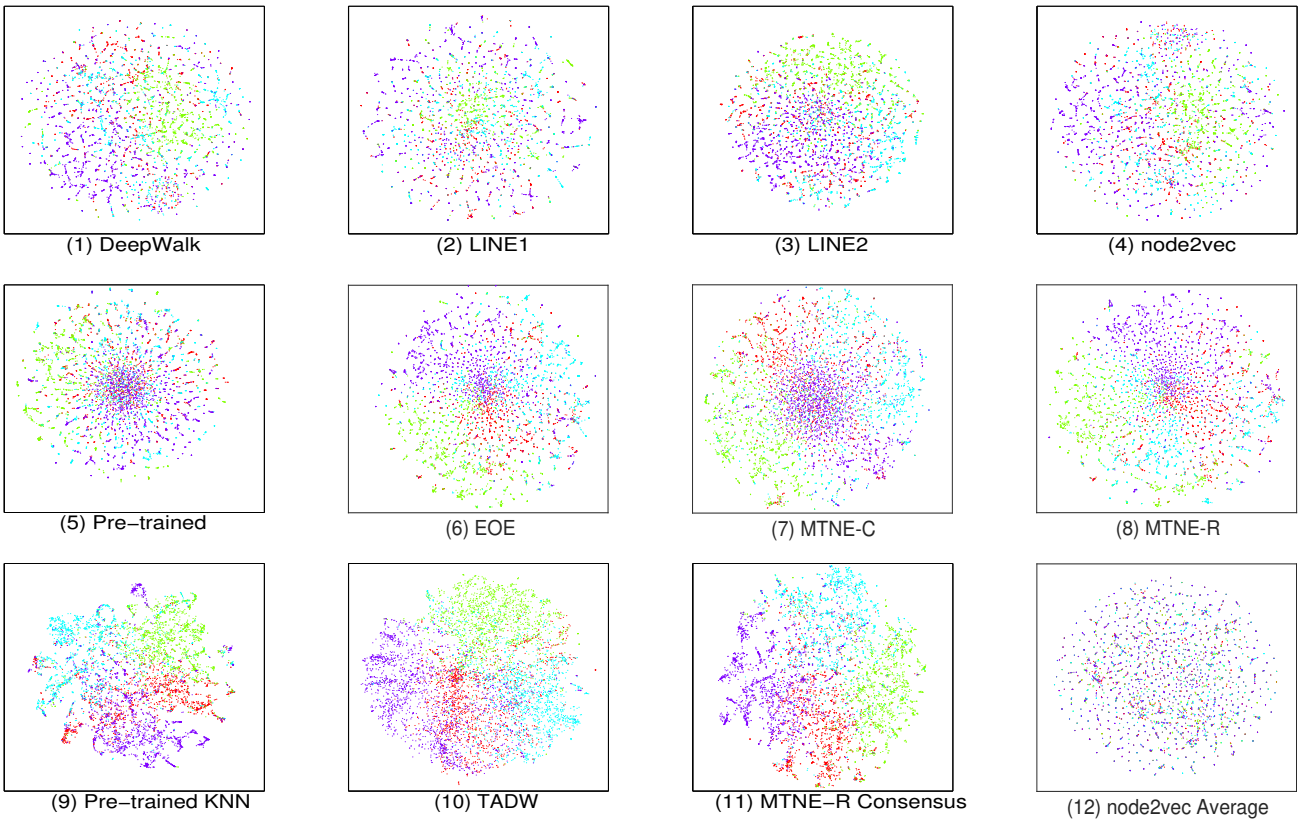


Fig. 2: Visualizations of embeddings learned from the DBLP co-authorship network by different models from (1) to (8), where different colors denote different research fields of researchers. Red color corresponds to DB, blue color corresponds to DM, green color corresponds to ML, and light blue corresponds to IR. (9) is visualization of pre-trained embeddings learned from the DBLP KNN network. (10) and (11) are visualizations of consensus embeddings of MTNE-R and common embeddings of MTNE-C, respectively. (12) is the average of three network-specific embeddings learned by node2vec.

### *B. Experiment Settings*

The five representative network embedding models mentioned in the related work are used as baselines, i.e., DeepWalk [19], LINE [22], TADW [31], node2vec [14], and EOE [30]. 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 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.

In all the evaluations, network-specific embeddings learned by MTNE-C are evaluated after being combined with corresponding common embeddings because neither network-specific embeddings nor the common embeddings are complete as indicated in the closeness measurement by Eq. (6). Correspondingly, all the network-specific embeddings learned by MTNE-R and the consensus embeddings are complete embeddings, and they are not combined for the evaluation.

### *C. Network Visualization*

This section presents visual evaluation of embeddings in a two-dimension 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 [16] to project them to 2-dimension vectors. The visualization results are presented in

| | DeepWalk | LINE(1st) | LINE(2nd) | node2vec | TADW | EOE | MTNE-C | MTNE-R | MTNE-C Common | MTNE-R Consensus |
|---|---|---|---|---|---|---|---|---|---|---|
| Co-authorship | 74.07 | 62. 39 | 75.36 | 71.07 | 56.25 | 76.69 | 78.29 | 77.62 | 56.81 | **78.92** |
| Citation | 83.97 | 80.32 | 84.88 | 84.66 | 58.62 | 85.36 | 86.54 | 85.80 | 61.02 | **87.84** |

TABLE II: AUC scores for link prediction tasks on the co-authorship network and the citation network, where all scores have been multiplied by 100%.

Fig. 2. Only visualizations of embeddings learned from the co-authorship network are presented because visualizations of embeddings from other two networks show similar patterns.

From Fig. 2(1) to Fig. 2(5) 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 MTNE-C and MTNE-R perform better by utilizing information of keywords of their papers which are usually domain-specific. Domain-specific keywords can be demonstrated by Fig. 2(9) where the visualization of pre-trained embeddings learned from the DBLP KNN network show 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) with final embeddings in Fig. 2(7) and Fig. 2(8).

TADW in Fig. 2(10) 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 $128 \times 2$, 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. However, the consensus embeddings of MTNE-R in Fig. 2(11) is 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. (17), 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. (12), and the average of other baselines are 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.

### *D. 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 [15]. 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 [14]. 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.

The AUC scores for tasks on the co-authorship network and the citation network are presented in Table 2. It shows that MTNE models perform better than all the baselines. MTNE 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. That utilizing more related information is beneficial can also be seen in the performance of EOE, which underperforms MTNE models because it can only utilize two sources of information. It seems to be strange that TADW performs very poor even though it utilizes two sources of information. We notice that TADW explicitly uses embeddings of keywords to measure the similarity. In this way, 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.

Network-specific embeddings learned by MTNE-C performs better than those learned by MTNE-R, which may be because the common characteristics are explicitly combined with network-specific embeddings in each task in MTNE-C while they are less explicitly incorporated in network-specific embeddings by making network-specific embeddings agree on the consensus embeddings.

One interesting observation is that the consensus embeddings learned by MTNE-R perform the best in both co-authorship prediction and citation prediction. Recall that the consensus embeddings summarize all the network-specific embeddings. Then the observation suggests that overall knowledge perform better than network-specific knowledge. We notice that the co-authorship network, 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 MTNE-C alone perform the worst because they only contain partial information, and are omit in the rest of evaluation.

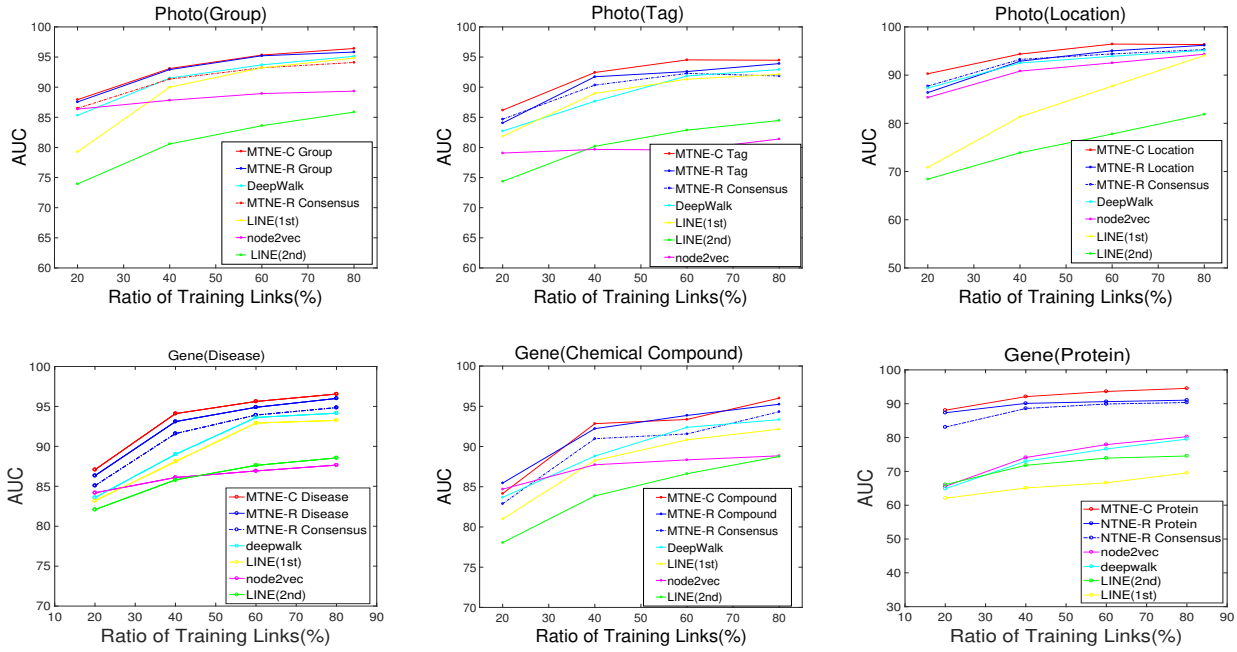


Fig. 3: AUC plots on link prediction tasks, where methods are placed according to its ranking of performance in the legend.

| | Model | LINE(1st) | | LINE(2nd) | | DeepWalk | | node2vec | | TADW | | EOE | | MTNE-C | | **MTNE-R** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | short | long | short | long | short | long | short | long | C1 | C2 | C1 | C2 | short | long | short | long |
| DBLP | Micro-F1 | 66.6 | 65.8 | 72.1 | 69.0 | 73.7 | 72.0 | 72.0 | 69.2 | 81.2 | 82.0 | 79.0 | 80.4 | 84.2 | 84.3 | 84.5 | **86.2** |
| | Macro-F1 | 63.6 | 62.8 | 67.9 | 67.1 | 68.7 | 67.2 | 67.1 | 64.5 | 73.1 | 73.5 | 72.2 | 72.9 | 75.2 | 75.1 | 75.8 | **78.4** |
| Flickr | Micro-F1 | 61.6 | 59.9 | 64.8 | 63.2 | 65.4 | 64.9 | 65.5 | 65.2 | – | – | – | – | 66.1 | 65.1 | **67.2** | 67.1 |
| | Macro-F1 | 61.1 | 59.6 | 64.2 | 62.7 | 65.1 | 64.7 | 65.1 | 64.7 | – | – | – | – | 65.9 | 64.9 | **67.1** | 66.9 |

TABLE III: Micro-F1 and Macro-F1 scores for multi-label classification, where numbers have been multiplied by 100%, C1 and C2 refer to co-authorship and citation, respectively.

For photo networks and 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 its ranking of performance. Similarly, the proposed MTNE 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.

Unlike in Table 2 where the consensus embeddings perform the best, network-specific embeddings learned by MTNE-C obtain the best performance, and network-specific embeddings learned by MTNE-R usually perform better than the consensus embeddings. Network-specific embeddings learned by MTNE-C perform better than those learned by MTNE-R may also be because the common characteristics are more explicitly combined with network-specific embeddings in each task in MTNE-C than in MTNE-R. Network-specific embeddings learned by MTNE-C perform even better than the consensus embeddings in that the three photo networks and 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.

To sum up, the consensus embedding learned by MTNE-R may be superior to network-specific embeddings in link prediction tasks where the multiple networks are closely related in the sense that all the links are established between nodes sharing the same characteristics. Otherwise, network-specific embeddings learned by MTNE-C are superior.

### *E. 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 [20] 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, MTNE 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 3. It shows MTNE-R 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 MTNE-R 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 visulizations illustrated in Fig. 2(11).

MTNE models outperform TADW and EOE because MTNE models 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 MTNE models may be even larger.

### *F. Comparison of MTNE-C and MTNE-R*

The only difference between MTNE-C and MTNE-R is the mechanism for incorporating an information-sharing embedding. MTNE-C incorporates the information-sharing embedding as a common embedding shared by all tasks in a similar way to multi-task metric learning [18]. Hence, the common embeddings capture the commonalities of a node across all networks, and are combined with a network-specific embedding to be a complete embedding. The advantage of this mechanism is that the commonalities are effectively captured and utilized in each task. As a result, MTNE-C usually has

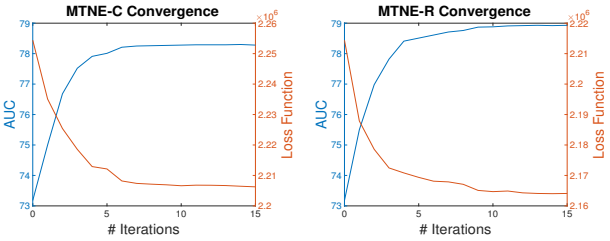


Fig. 4: Convergence analysis

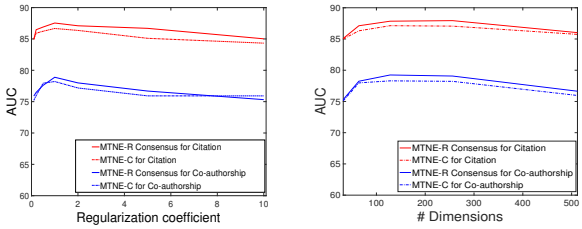


Fig. 5: Parameter analysis, where network-specific embeddings are used in the tasks for MTNE-C while the consensus embeddings are used for MTNE-R.

the better performance in network-specific applications, such as link prediction illustrated in Fig. 3.

MTNE-R incorporates the information-sharing embedding as a consensus embedding on which all network-specific embeddings agree. In other words, the consensus embedding summarizes all the network-specific embeddings, which is the advantage of MTNE-R. As a result, MTNE-R usually has the better performance in applications which require overall information, such as visualization and multi-label classification. However, there is a special case where MTNE-R may outperform MTNE-C in network-specific applications as illustrated in link prediction for the DBLP dataset. In this special case, the links of all the networks are established between nodes sharing the same properties, e.g., research interests for the DBLP dataset.

### *G. 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 because other experiments show similar results. The performance of MTNE-C is obtained by network-specific embeddings while the performance of MTNE-R 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.

### *H. Parameter Sensitivity*

This section evaluates how the proposed models are sensitive to the regularization coefficients and the dimension of

embeddings. The performance of link prediction on the DBLP networks 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 be within the range [0.05, 2], and the dimension should better be within the range [64, 256];

## IX. Conclusion and Future Work

This paper proposes for the first time to jointly embed multiple networks which share the same set of nodes. As the same node may expose similar or complementary characteristics across multiple networks, jointly embedding multiple networks can make each network-specific embeddings more comprehensive and complete. 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 MTNE-C and as a consensus embedding in MTNE-R. Through comprehensive experiments, we demonstrate the proposed MTNE models outperforms state-of-the-art network embedding models in applications including visualization, link prediction, and multi-label classification. In the future, we plan to enable MTNE models to perform online learning so as to handle evolving networks.

## Acknowledgment

The work described in this paper was partially supported by the funding for Project of Strategic Importance provided by The Hong Kong Polytechnic University (Project Code : 1-ZE26), the funding for demonstration project on large data provided by The Hong Kong Polytechnic University (project account code : 9A5V), and NSF through grants IIS-1526499, and CNS-1626432, and NSFC 61672313.

## References

[1] Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

[2] Bart Bakker and Tom Heskes. Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research*, 4(May):83–99, 2003.

[3] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.

[4] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.

[5] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines*, pages 567–580. Springer, 2003.

[6] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[7] James C Bezdek and Richard J Hathaway. Some notes on alternating optimization. In *AFSS International Conference on Fuzzy Systems*, pages 288–300. Springer, 2002.

[8] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. Node classification in social networks. In *Social network data analytics*, pages 115–148. Springer, 2011.

[9] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[10] Bin Chen, Ying Ding, and David J Wild. Assessing drug target association using semantic linked data. *PLoS Comput Biol*, 8(7):e1002574, 2012.

[11] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[12] Trevor F Cox and Michael AA Cox. *Multidimensional scaling*. CRC press, 2000.

[13] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM, 2004.

[14] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[15] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[16] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[17] Julian McAuley and Jure Leskovec. Image labeling on a network: using social-network metadata for image classification. In *European Conference on Computer Vision*, pages 828–841. Springer, 2012.

[18] Shibin Parameswaran and Kilian Q Weinberger. Large margin multi-task metric learning. In *Advances in neural information processing systems*, pages 1867–1875, 2010.

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

[20] Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. Meka: a multi-label/multi-target extension to weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016.

[21] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[22] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. ACM, 2015.

[23] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM, 2008.

[24] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[25] Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.

[26] Xiaokai Wei, Bokai Cao, Weixiang Shao, Chun-Ta Lu, and Philip S. Yu. 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*, pages 773–782, 2016.

[27] Xiaokai Wei, Sihong Xie, and Philip S. Yu. 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*, pages 82–90, 2015.

[28] Xiaokai Wei, Linchuan Xu, Bokai Cao, and Philip S. Yu. 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*, pages 1611–1619, 2017.

[29] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S. Yu. 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*, pages 859–860, 2017.

[30] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S Yu. Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 741–749. ACM, 2017.

[31] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. Network representation learning with rich text information. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.