Hajiseyedjavadi *et al. Applied Network Science* (2019) 4:125
https://doi.org/10.1007/s41109-019-0242-0

Applied Network Science

RESEARCH Open Access

Learning embeddings for multiplex networks using triplet loss

Seyedsaeed Hajiseyedjavadi*, Yu-Ru Lin and Konstantinos Pelechrinis

*Correspondence: seh138@pitt.edu
University of Pittsburgh, Pittsburgh, PA USA

Abstract

Learning low-dimensional representations of graphs has facilitated the use of traditional machine learning techniques to solving classic network analysis tasks such as link prediction, node classification, community detection, etc. However, to date, the vast majority of these learning tasks are focused on traditional single-layer/unimodal networks and largely ignore the case of multiplex networks. A multiplex network is a suitable structure to model multi-dimensional real-world complex systems. It consists of multiple layers where each layer represents a different relationship among the network nodes. In this work, we propose MUNEM, a novel approach for learning a low-dimensional representation of a multiplex network using a triplet loss objective function. In our approach, we preserve the global structure of each layer, while at the same time fusing knowledge among different layers during the learning process. We evaluate the effectiveness of our proposed method by testing and comparing on real-world multiplex networks from different domains, such as collaboration network, protein-protein interaction network, online social network. Finally, in order to deliberately examine the effect of our model's parameters we conduct extensive experiments on synthetic multiplex networks.

Keywords: Multiplex network, Network embedding, Triplet loss

Introduction

Networks offer a rich way to represent a large number of phenomena and relationships between variables of interest. While traditional network science research has led to the development of a variety of analytical tools that can provide us with a detailed view of the structure and processes that take part over a network, until recently powerful machine learning methods were not able to be (fully) utilized in network analysis. Given this prevalence of networked data and to allow for the application of machine learning techniques in network analysis, there has been a growing interest in learning low-dimensional representations of graphs. These representations, essentially *vectorize* nodes (or edges) of the network, which further allows to apply state-of-art machine learning algorithms to tackle network-related tasks such as community detection, link prediction, network similarity and even graph visualization.

These methods can be broadly classified into three main categories, namely, (**a**) matrix factorization (**b**) random walk-based, and (**c**) neural network-based. The first category tries to reduce the dimensions of the adjacency matrix so that the resulting low-ranked matrix keeps the existing relationships of the original adjacency matrix. In this category

we can refer to Singular Value Decomposition (SVD) (Ou et al. 2016) and Non-negative matrix factorization (Wang et al. 2017a). In the second class, different types of random walks are utilized to generate samples of nodes that preserve some local (or global) structure and then they are used to learn a vector representation through a suitable optimization problem (Perozzi et al. 2014; Grover and Leskovec 2016). Approaches belonging to the third category utilize artificial neural networks to obtain latent representation of nodes. The key aspect in these approaches is to obtain samples of node pairs that are *similar* (positive samples), and hence, need to be close in the latent space to be learned, and node pairs that are not similar (negative samples) (Wang et al. 2017b).

The vast majority of the existing methods for learning low-representations of network objects operates on single-layer, unimodal, networks. That is, networks that represent a single type of relationship between the node(Grover and Leskovec 2016; Perozzi et al. 2014). However, in many real-world systems there are several relationships between the network nodes that need to be captured. For example, there are several relationships between people living in a neighborhood, such as, friendship, professional, blood relationships etc. These relationships reflect different characteristics of the entities and hence it necessitate the use of a multidimensional model to represent these relationships distinctively. (Figure 1)

Attempting to represent these interactions by aggregating them into a single unimodal network leads to loss of valuable information (Aleta and Moreno 2018). Therefore, to

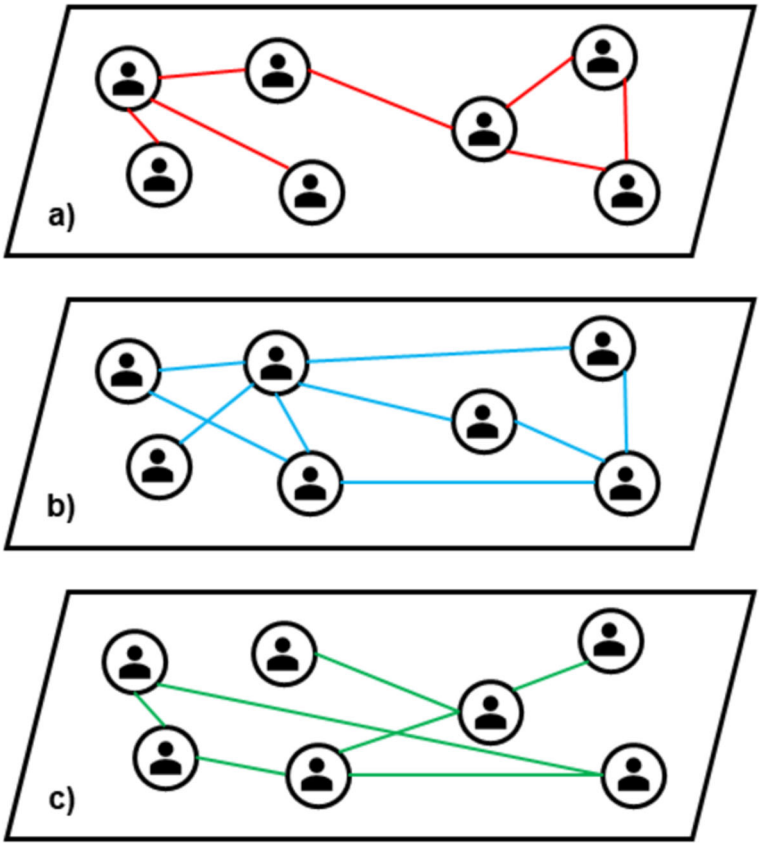

Fig. 1 An example of a multiplex network. Different layers represent different relationships

model these systems, multilayer networks are introduced. In a multilayer network, a network is built from a set of layers that each one describe different type of interactions or relationships between entities. Multiplex network is a specific type of multilayer networks where each layer represents the different relationships of the same set of nodes. These layers can also be interconnected capturing inter-dependencies between the different layers. In multiplex network, if there is any interlayer link, it connects counterpart nodes located on different layers. Examples are but not limited to proteins-protein interaction network (Zitnik and Leskovec 2017) in a specific human tissues in which tissue-specific gene-function relationships between layers exists or a transportation network that consists of layers representing different modes of transportation . In latter, passengers could switch between modes of transportation which could be modeled by interlayer links (Strano et al. 2015). Motivated by the importance of this type of network, recently, a few papers have addressed the problem of learning representation of nodes in a multiplex network. Liu et al. (2017a) extended the idea of Node2vec (Grover and Leskovec 2016) to multiplex networks. In this approach, the random walker not only can traverse on nodes of a layer, but also based on the value of a jumping factor, it can be transported to the the same node in another layer of the network. At the end, they would have the numerous sequences of visited nodes generated from all layers that can be used as the input to softmax classifier used in Node2vec. Also, Zhang et al. (2018) introduce two different embeddings for nodes in a multiplex network. One is a high-dimensional common embedding and a low-dimensional embedding vector for each layer of the network. The final embedding of a node is the linear aggregation of these two vectors.

As alluded to above, most of the existing low-dimensional representation learning approaches operate on single-layer networks, but do not necessarily perform well on multiplex networks. While for each one of these different *layers* we can use one of these off-the-shelf network representation learning (embedding) algorithms to identify representations of the nodes, this largely ignores information from the rest of the layers, which can be valuable. In this work, we propose MUNEM a novel approach toward MUltiplex Network EMbedding using a triplet loss objective function. In MUNEM, at each step, we create a triplet of nodes consists of randomly selected anchor node and comparing it with both a positive and a negative sample. The distance between the anchor node and the negative one must be higher than the distance between the anchor node and the negative node. This objective function gives us the flexibility to both learn the structure of each single independently, and pair nodes between layers to fuse the learned knowledge across layers. We test our method with other baseline methods to predict links exists between nodes on a single layer of the network. The experiments depict that MUNEM outperforms all of the baseline methods in all of the data sets.

The rest of the paper is organized as follows: "Related work" section discusses related to our study literature. "Proposed methodology" section presents in detail our proposed method. Then, in "Experiments" section we evaluate the effectiveness of our proposed method by testing and comparing with baselines on synthetic networks as well as real-world networks gathered from different domains.

Related work

In this section, we briefly survey notable related works addressing the methodological problem of learning low-dimensional representations on different types of networks.

Network embedding

Network embedding is a prominent research area that its goal is to learn low-dimensional representation of nodes in a graph. The learned latent vectors are expected to preserve the structure and any existing attributes of the input network. Although, there has been some classic approaches like (Belkin and Niyogi 2002; Tenenbaum et al. 2000), they were only applicable on capturing local structure of small or medium sized networks. Recently, inspired by a work of Mikolov et al. (2013) in which they introduced the Skip-gram model for learning low-dimensional representations of words in a huge corpus, a new generation of scalable methodologies were proposed in the recent years. Inspired by their work, Deepwalk (Perozzi et al. 2014) and Node2vec (Grover and Leskovec 2016) were introduced to use similar approach for learning representations on graphs. These methods consider nodes that are visited by a random walker analogues of words appearing in a sentence. Generating a large corpus of words, i.e. the visited nodes, they employed the very similar Skip-gram model to learn representation of each node. However, the latter one, provide a more flexible sampling strategy that controls the random walker whether to traverse in the neighborhood of the recently visited node(s) or sample nodes from a deeper structure of the network. Another approach named LINE was introduced by Tang et al. (2015) to capture first-order and second-order proximity of nodes.

In recent years, various number of methodologies were proposed to learn low-dimensional representations of different type of networks. In Wang et al. (2017b), the authors provide a deep learning framework to learn node embeddings of a signed social network. They design an objective function that ensures “friends" have closer representations than “foes". A network with various node types, i.e. heterogeneous network, is another interesting topic which was addressed in Chang et al. (2015). The authors have a network which its nodes are images and texts and therefore three types of connections. Using a deep neural network structure they learn the embeddings of each type then, the embeddings are mapped to a common space by using a linear layer. In case of network embedding with side information, we can refer to (Le and Lauw 2014) in which they introduce a generative model to learn the embedding of a network of documents. In this network, each node is a document which not only contains valuable content, but also structurally connected to other nodes.

Multiplex network

Many real-world systems are characterized by different views and properties that can not be modeled by single network. Multiplex networks are designed as more advanced network structures to capture various existing interactions or relationship between the same entities in a complex system. In a Multiplex network, each type of relationship between nodes is represented through set of edges connecting nodes in a layer. Temporal dynamic networks is another type of network that could be modeled by Multiplex network in the way that snapshots of the network taken in a specific time-window is considered as layers (Javadi et al. 2018).

In recent years, Zhang et al. (2018) propose a method to learn two different representations for nodes in a multiplex network. One high-dimensional vector which is identical for each node across different layers and a low-dimensional embedding vector which is being learned for each relation (i.e. layer) during the learning process. A node’s final

representation is linear summation of the shared embedding vector and the low-dimensional vector specifically learned for for each layer. Since these two vectors have different sizes, the authors introduce a transformation matrix to align these two vectors into the same size. In Ohmnet (Zitnik and Leskovec 2017), the authors focus on a multiplex network in which layers represent molecular interactions in a different human tissue. Using a rich multi-scale tissue hierarchy, they enforce sharing of similar features between nodes of similar network neighborhoods, i.e. proteins activated in similar tissues. Although they have reached a higher accuracy in predicting of tissue-specific cellular functions, their method is hard to be generalized for other datasets lacking hierarchical similarities of layers. Another framework named MINES (Ma et al. 2018) was proposed to learn embedding space of nodes in a multi-dimensional e-commerce network. Their learning methodology consists of independent structure of each layer and shared information of nodes between layers. For the latter one, they utilize the categories of nodes as a mean to capture dependent information of the nodes with the same attribute located on different layers. However, without the attribute information of nodes, their method loses the shared information and simply learns representations of nodes on different layers independently.

Proposed methodology

In this section, we present our proposed method. First, we introduce the notations used throughout the paper. Then, after we describe our single layer embedding approach, we provide details of our proposed method to learn low-dimensional representations of a multiplex network.

Notations and definitions

We model a multiplex network G as a set of L network layers $\{g^1, g^2, ..., g^L\}$, where $g^l = (V^l, E^l)$ denotes a specific layer consisting of the set of nodes V^l and E^l intra-layer edges. In our model, the set of nodes are identical on all layers. Therefore, as long as a node is not isolated on at least on one layer, we include that in all layers. The set of all nodes on all layers is represented as $\mathcal{V} = \{V^1, V^2, ..., V^L\}$, and the set of intra-layer edges $\mathcal{E}$ is defined as a set of all edges on all layers, i.e. $\mathcal{E} = \{E^1, E^2, ..., E^L\}$.

The goal of our work is to find a mapping from V_i^l to $f_i^l \in \mathbb{R}^s$ where $s << |\mathcal{V}|$. In this work, our objective is to learn low-dimensional representations of nodes of each layer in a way that not only it preserve network structure of each layer, but also incorporate rich information of other layers in the learning process. We use notion of f_i^l to denote the learned embedding space of node i on layer l.

Single-Layer network embedding

In this section, we introduce our basic embedding methodology that is applicable on a single layer network. In our work, we employ triplet loss objective function which had a remarkable performance in the task of face recognition and clustering (Schroff et al. 2015). The triplet loss objective function strives to learn representation in the way that the square distances between entities “related" to each other be less than the square distances of entities that are “non-related". The terms “related" and “non-related" could be interpreted differently in various context. As an instance, in Schroff et al. (2015), all images of a specific person are considered to be related and all pairs of images of non-identical

persons are considered as non-related. Hence, the objective function encourages that feature space of an image taken from a specific person encouraged to be closer to another image of the same person rather than an image of any other one.

For our single-layer network embedding we use the first-order proximity of nodes to define similar and non-similar entities, that is, if two nodes are connected through an edge they are considered to be similar, otherwise they are not similar. From the point of a given node V_a^l (*anchor*) located on layer l, we select a node V_p^l (*positive*) from its neighbors and another node V_n^l (*negative*) from the set of non-neighbor nodes (Fig. 2a). To sample the triplets, for every link, we choose one endpoint of the link as anchor and the other one as positive. Then, for the negative, we only need to randomly select a node not connected to the anchor node. For large networks containing millions of links this might not computationally effective to sample all links, therefore, we could sample a portion of links for the sake of learning process.

By collecting N possible triplets from layer l, we need to minimize the following objective function:

$$L^{(l)} = \sum_{\forall\left(V_a^l, V_p^l, V_n^l\right) \in N^l} max\left(0, ||f_a^l - f_p^l||_2^2 - ||f_a^l - f_n^l||_2^2 + \delta_{in}\right) \quad (1)$$

where f_a^l, f_p^l and f_n^l denote the representations of anchor node V_a^l, positive sample V_p^l and the negative sample V_n^l, respectively.The set N^l contains the triplets sampled from layer l. The hyper-parameterδ_{in} enforces a margin between the distances of positive and negative pairs.

In MUNEM, this objective function enforces that the squared distances of representations of two connected nodes need to be less than squared distances of representations of two non-connected nodes. Although we choose first-order proximity as criteria of similarity, one can extend it to higher-order proximity as well.

To compare our negative sampling strategy with Node2vec(Grover and Leskovec 2016) we must explain that in node2vec a negative node is randomly drawn from the noise distribution. It is a distribution of nodes based on how frequently it is appeared in the sequence of nodes visited by a random walker. Without any checking process in use, it is likely to have pairs of nodes both as negative and positive samples. It usually happens when two nodes appeared in a sequence both have high degree centralities. Two nodes are

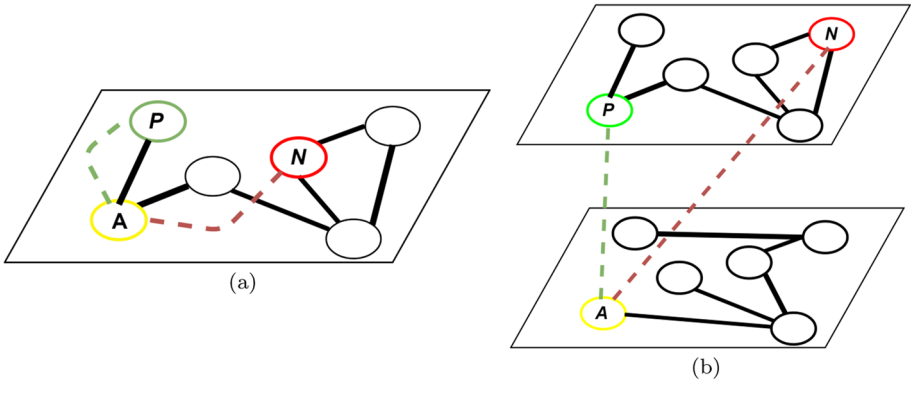

Fig. 2 A toy network to illustrate in-layer triplet sampling (left) and inter-layer triplet sampling (right)

most likely to be appeared in the same sequence when they are both connected or there are a high relative number of common neighbors existing between them . This structure is very common in real-world network (Valente et al. 2008) and it causes to decrease the effectiveness of the method. The other drawback of this approach is that the noise distribution need to be made after all sequence of visited nodes are generated which it requires to traverse all nodes of the network. In contrast, in the MUNEM, we can generate triplets of a node only by looking at the local information (i.e. neighborhood) of that specific node to pick a positive from its neighbors and negative from the non-connected ones.

Multiplex network embedding

In order to fuse knowledge of layers into the learning process, we sample triplets from different layers. In our approach, same nodes on different layers are encouraged to have closer representations than non-identical nodes located on different layers. This essentially allows us to fuse information obtained from different dimensions. To do so, for any non-isolated anchor node V_a^l located on layer l we sample the same node of another layer V_a^k , where $k \neq l$, as a positive pair and a random non-isolated node V_n^k from the set of nodes not connected to V_a^k as a negative pair (Fig. 2b). By collecting M possible triplets we need to minimize the following objective function:

$$Q = \sum_{\forall (V_a^l, V_a^k, V_n^k) \in M} max\left(0, ||f_a^l - f_a^k||_2^2 - ||f_a^l - f_n^k||_2^2 + \delta_{bet}\right) \tag{2}$$

where f_a^l, f_a^k and f_n^k denote the representations of anchor node V_a^l, positive sample V_a^k and the negative sample V_n^k, respectively. This objective function enforces the distances between the counterpart nodes on different layers, V_a^l and V_a^k, be less than the distances between V_a^l and V_n^k by the margin of δ_{bet}.

In order to both fuse knowledge of within and between layers we introduce the following objective function:

$$P = Q + \sum_{\forall l \in L} L^{(l)} \tag{3}$$

By minimizing P not only we do preserve the global structure of each layer, but we also fuse knowledge between different layers during the learning process by pairing counterpart nodes on different layers.

Optimization

To optimize the aforementioned objective function, we use Stochastic Gradient Descent as an efficient approach to learn the embeddings. Here, we present the gradient needed to update every parameters of the method.

$$\frac{\partial Loss}{\partial f_a^l} = \sum_{\forall (V_a^l, V_p^l, V_n^l) \in N^l} \begin{cases} 2\left(f_n^l - f_p^l\right), \ if \ \left(||f_a^l - f_p^l||_2^2 - ||f_a^l - f_n^l||_2^2 + \delta_{in}\right) \geq 0 \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{4}$$

$$\frac{\partial Loss}{\partial f_p^l} = \sum_{\forall (V_a^l, V_p^l, V_n^l) \in N^l} \begin{cases} -2\left(f_a^l - f_n^l\right), \ if \ \left(||f_a^l - f_p^l||_2^2 - ||f_a^l - f_n^l||_2^2 + \delta_{in}\right) \geq 0 \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{5}$$

$$\frac{\partial Loss}{\partial f_n^l} = \sum_{\forall (V_a^l, V_a^k, V_n^k) \in N^l} \begin{cases} 2\left(f_a^l - f_p^l\right), \ if \ \left(||f_a^l - f_p^l||_2^2 - ||f_a^l - f_n^l||_2^2 + \delta_{in}\right) \geq 0 \\ 0, \qquad\qquad otherwise \end{cases} \tag{6}$$

For between layers learning we would have :

$$\frac{\partial Loss}{\partial f_a^l} = \sum_{\forall (V_a^l, V_a^k, V_n^k) \in M} \begin{cases} 2\left(f_n^k - f_a^k\right), \ if \ \left(||f_a^l - f_a^k||_2^2 - ||f_a^l - f_n^k||_2^2 + \delta_{bet}\right) \geq 0 \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{7}$$

$$\frac{\partial Loss}{\partial f_a^k} = \sum_{\forall (V_a^l, V_a^k, V_n^k) \in M} \begin{cases} -2\left(f_a^l - f_n^k\right), \ if \ \left(||f_a^l - f_a^k||_2^2 - ||f_a^l - f_n^k||_2^2 + \delta_{bet}\right) \geq 0 \\ 0, \qquad\qquad otherwise \end{cases} \tag{8}$$

$$\frac{\partial Loss}{\partial f_n^k} = \sum_{\forall (V_a^l, V_a^k, V_n^k) \in M} \begin{cases} 2\left(f_a^l - f_a^k\right), \ if \ \left(||f_a^l - f_a^k||_2^2 - ||f_a^l - f_n^k||_2^2 + \delta_{bet}\right) \geq 0 \\ 0, \qquad\qquad Otherwise \end{cases} \tag{9}$$

In Algorithm 1 we summarize our approach.

Algorithm 1: MUNEM Algorithm

Data: Multiplex network which is composed of network layers $\{g^1, g^2, ..., g^L\}$, embedding dimension d, learning rate η, margin parameters for in-layer embedding δ_{in}, margin parameters for between layers embedding δ_{bet},

Result: Low-dimensional representations for different nodes on each layer f_i^l

initialization: Initialize weights of nodes uniformly distributed

[0.5 cm] For each layer l we make the set N^l of in-layer triplets

Make the set M consists of between layer triplets

Create a set O from merging two sets of samples N^l and M

while *not converged* **do**

calculate loss by Eq 3

select randomly a triplet from the set O as $\left(V_a^l, V_p^k, V_n^k\right)$

if k==l **then**

do

$$f_a^l = f_a^l - \eta \frac{\partial Loss}{\partial f_a^l}$$

$$f_p^l = f_p^l - \eta \frac{\partial Loss}{\partial f_p^l}$$

$$f_n^l = f_n^l - \eta \frac{\partial Loss}{\partial f_n^l}$$

else

do

$$f_a^l = f_a^l - \eta \frac{\partial Loss}{\partial f_a^l}$$

$$f_a^k = f_a^k - \eta \frac{\partial Loss}{\partial f_a^k}$$

$$f_n^k = f_n^k - \eta \frac{\partial Loss}{\partial f_n^k}$$

end

end

return f_i^l

Parameter boundaries

It is essential to note that in order to constrains the representations to be on a hypersphere, L2 normalization is applied on the learning representations. This guarantees that representation embedded in a Euclidean space and therefore an angular distance metric is

applicable to the learned space (Liu et al. 2017b). The L2 normalization also restrict that the range of the effective margin parameter δ_{in} to be in the interval of [0,4]:

$$||\frac{A}{||A||} - \frac{P}{||P||}||_2^2 - ||\frac{A}{||A||} - \frac{N}{||N||}||_2^2 \tag{10}$$

$$= ||\frac{A}{||A||}||_2^2 - 2\frac{A \cdot P}{||A||||P||} + ||\frac{P}{||P||}||_2^2 - ||\frac{A}{||A||}||_2^2 + 2\frac{A \cdot N}{||A||||N||} - ||\frac{N}{||N||}||_2^2 \tag{11}$$

$$= 1 - 2\frac{A \cdot P}{||A||||P||} + 1 - 1 + 2\frac{A \cdot N}{||A||||N||} - 1 \tag{12}$$

$$= 2\left(\frac{A \cdot N}{||A||||N||} - \frac{A \cdot P}{||A||||P||}\right) \tag{13}$$

and since the value of the cosine similarity of two vectors ranges between -1 and 1 we would have:

$$-4 \leq \left(||\frac{A}{||A||} - \frac{P}{||P||}||_2^2 - ||\frac{A}{||A||} - \frac{N}{||N||}||_2^2\right) \leq 4, \tag{14}$$

therefore, in order to keep the value of the equation positive, the margin value only needs to be chosen as a value in the range of [0,4].

Experiments

To evaluate the effectiveness of our method, we examine it on both real-world and computer-generated multiplex networks.

Datasets

Real-world networks

We select four real-world networks from various domains with sizes ranging from less than hundred to hundreds of thousands nodes. In these networks, each node has at least one link on one layer. In other words, if there is a node that is isolated on all layers, we exclude it from the network. Here are the description of the networks we use in our experiments:

Celegans: This multiplex network consists of different synaptic junctions of genetic interactions (Chen et al. 2006).

Genetic: This is the multiplex genetic and protein interactions network of the Saccharomyces Pombe. In this network, multiple types of genetic interactions for organisms are included and represented as a separate layer. For more technical explanation on each type of interaction we refer you to (De Domenico et al. 2015; Stark et al. 2006).

ArXiv: It's a multiplex network consists of layers corresponding to different ArXiv categories. In this network, each node is a researcher and links represent the collaboration between researchers. To restrict the analysis to a well-defined topic of research, this dataset contains papers with "networks" in the title or abstract up to May 2014 (De Domenico et al. 2015).

Climate: It consists of different types of social relationships among Twitter users during a large-scale activist event to advocate global action against climate change which held in New York in 2014. The layers of this network are corresponding to retweet, mentions and replies between users (Omodei et al. 2015). Table 1.

Table 1 The properties of real-world networks used in our experiments

Dataset	Number of nodes	Number of edges	Number of layers
Celegans	279	5,863	3
Genetic	4,092	63,677	7
ArXiv	14,488	59,026	13
Climate	102,439	348,973	3

Synthetic networks

In spite of real-world datasets, we examine the effectiveness of our method, by comparing its performance with the baselines on artificially generated multiplex networks. This is due the fact that by changing parameter values we would be able generate a wide range networks with different levels of modularity. Modularity is one of the most important characteristics of a complex network that measures how well subset of nodes are densely interconnected to each others and loosely connected to other subset of nodes. Real-world networks have different levels of modularity (Leskovec et al. 2009) and hence to represent a wide range of real-world examples we generate multiplex networks with different levels of modularity. First, we need to generate multiple layers of networks that each of which has characteristics of a scale-free networks. To do so, we generate a single network of LFR benchmark(Lancichinetti et al. 2008). A network is generated by specifying exponents of power-law distributions for both edge degree and community sizes and the mixing parameter μ ranges between 0 to 1 which set the average fraction of neighbors of a node that belongs to different communities that the node belongs to. A synthetic multiplex network with L layers would be made by making $L-1$ copies of the generated network. Then, for each layer network layers, we do the edge sampling by keeping each link with the probability of $1/L$.

We test the effectiveness of our method on different set of synthetic networks with 4 layers and default value suggested in the benchmark except the mixing parameters μ. We change the mixing parameter to produce networks with different structure.

Baseline methods

In our experiments, we compare our approach with the following methods:

Aggregated Deepwalk: The original DeepWalk employs a random walker to generate sample of nodes by traversing on the network. The generated sequence of visited nodes are treated as a sentence to use Skip-gram to learn the node embeddings. Since Deepwalk is an embedding method that works only on a single layer network, we apply Deepwalk on each layer independently, and then, for each node, we concatenate the embeddings of that node learned on each layer.

Aggregated Node2Vec: It takes the same steps of Deepwalk with this difference that two parameters are introduced to control the random walker movement. By specifying different values for these parameters, we encourage the random walker to stay in the neighborhood of currently visited node or traverse to farther nodes. Similar to the Aggregate Deepwalk, the learned embeddings of each layer will be concatenated, so each node would have one representation composed of independent representation of that node on different layers.

Principled Multilayer Network Embedding: Similar to DeepWalk and Node2Vec this method use random walk to generate a sequence of nodes. However, in this approach,

the random walker could jump between layers. We report the accuracy their method introduced in Liu et al. (2017a).

In addition to the aforementioned embedding methods, we will test our approach with the state-of-the-art network structure-based link prediction algorithms.

Jaccard Coefficient: For a pair of nodes, it measures the normalized number of common neighbors of two nodes by the number of their total neighbors.

Adamic Adar: This metric emphasizes on common neighbors with low degree by giving more weight to them.

In these link prediction algorithms, the high value of Jaccard coefficient or Adamic Adar index between a pair of nodes are interpreted as high probability of existence of a link connecting those nodes.

Evaluation metrics

Given a multiplex network, we report the accuracy of the link prediction task for one layer at a time. For a layer l, we split its links into training and test sets. After excluding positive test edges from that layer, the multiplex network is given to each method. The ratio of links chosen for the test set ranges from 10% to 90% in 10% increments. Accordingly, the higher ratio of edges are removed from the layer, the more probable the residual network would have higher number of connected components. For testing we sample uniformly non-neighbor node pairs as negative testing links as the same size of positive testing links. Both positive and negative test edges are used to evaluate the effectiveness of each method in predicting existence of a link between the pair of node on the layer l.

For embedding algorithms, after we learned the representations of nodes, we measure the cosine similarities of the node representations of layer l. The higher the cosine value, the more likely a link is connecting those nodes. After repeating this process for all layers, the average Area Under the Curve (AUC) as a standard metric for evaluating link prediction tasks, is reported. For the sake of a fair comparison, we use the same set of positive and negative links for our method and all other baselines.

Experimental settings

Model parameters

We followed default variables suggest in the original papers for methods needed parameter settings. For all the embedding based methods, we set their embedding dimensions and the dimension of our embedding to be 32. For Node2Vec, we empirically use p = 2 and q = 0.5. For MUNEM, we keep the same embedding length mentioned above. For two parameters δ_{bet} and δ_{in} we set values 1.5 and 1 for all experiments.

Result

Based on the experiment results, from the Fig. 3 we can observe that aggregated DeepWalk and aggregated Node2Vec perform poorly in compare to MUNEM and even PMNE. These methods, that are originally used on single networks, can not fuse knowledge between layers and consequently end up in loosing valuable information. This indicates that by learning each layers independently even with the state-of-the-art learning algorithms we may not learn the possibly complement information embedded through layers.

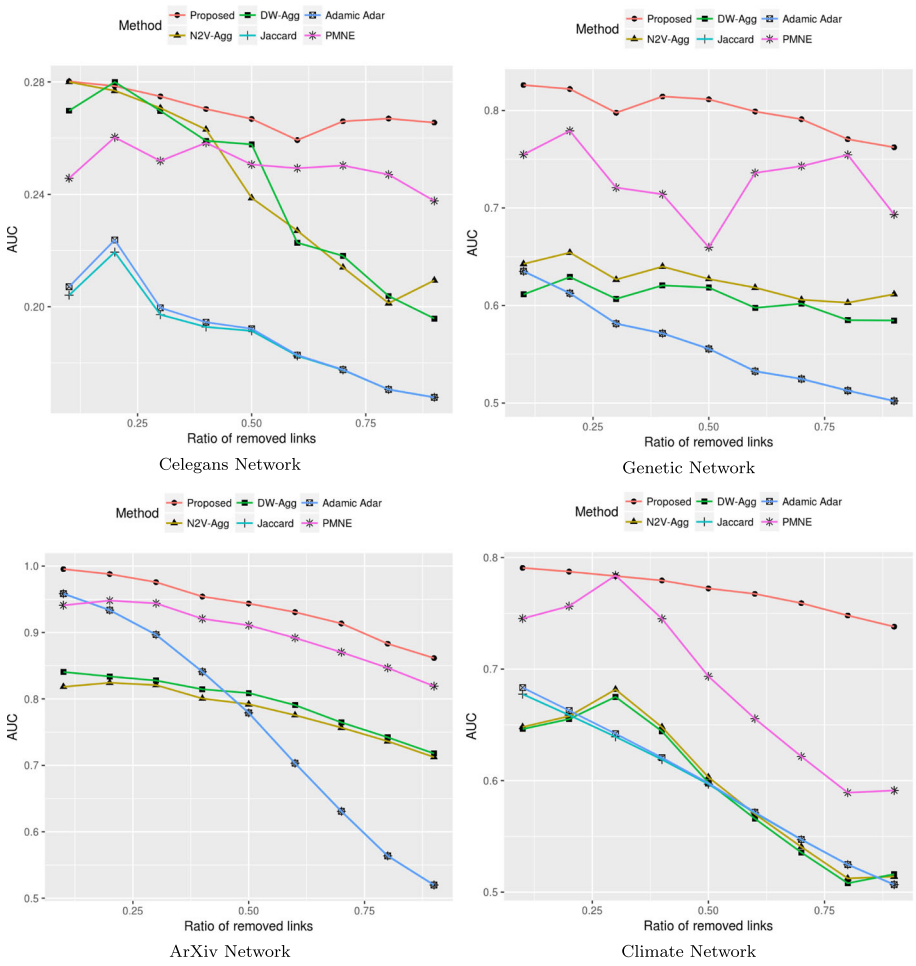

Fig. 3 Performance of the proposed model on different real-world networks

On the other hand, PMNE, which is roughly an extension to Node2Vec outperforms aggregated Node2vec, Adamic Ada and Jaccard link prediction methods, most of the time due the fact that the random walker they use in their method could be transported between layers. However, in all cases, MUNEM outperform PMNE by a significant margin, especially in the larger networks.

Another observation is that the more links we remove from a layer, the less link prediction algorithms, i.e. Adamic Adar and Jaccard are successful. This happens because of intrinsic approach of these methods that only consider the local information of nodes. Without any knowledge from global structure of the networks, they are highly vulnerable to missing links. In contrast, MUNEM by fusing information from different layers, compensate the missing information of a layer by incorporating embedding vectors of counterpart nodes of the other layers of the network.

To provide a stronger evidence for the effectiveness of our method, we generate synthetic network described in "Synthetic networks" section. These networks resemble the very main characteristics of real-world networks. In our experiment, we vary the mixing parameters to produce multiplex networks with different structure. The mixing

parameter μ ranges between 0 to 1 which define the average ratio of neighbors of a node that belongs to different communities that the node belongs to.

Again, as it can be seen in Fig.4, we observe the effectiveness of MUNEM against its competitors. It is worth mentioning that even with mixing parameter of 0.6, MUNEM could achieve a remarkable performance.

All in all, we test our models to state-of-the-art methods with different approaches. We observed that MUNEM outperforms the baselines in both real-world networks as well as synthetic networks. We believe that the process of pairing layers to fuse knowledge across them is the key element that helps this effectiveness.

Parameter sensitivity

We already provided the boundaries of both parameter δ_{in} and δ_{bet} in our model. Here in Fig.5, we report the AUC values we get when we use different values for these two parameters on Celegans network.

It can be seen that the sensitivity of the model to δ_{bet} is higher to δ_{in}. This observation, again, indicates the important role of fusing knowledge of layers in a multiplex network.

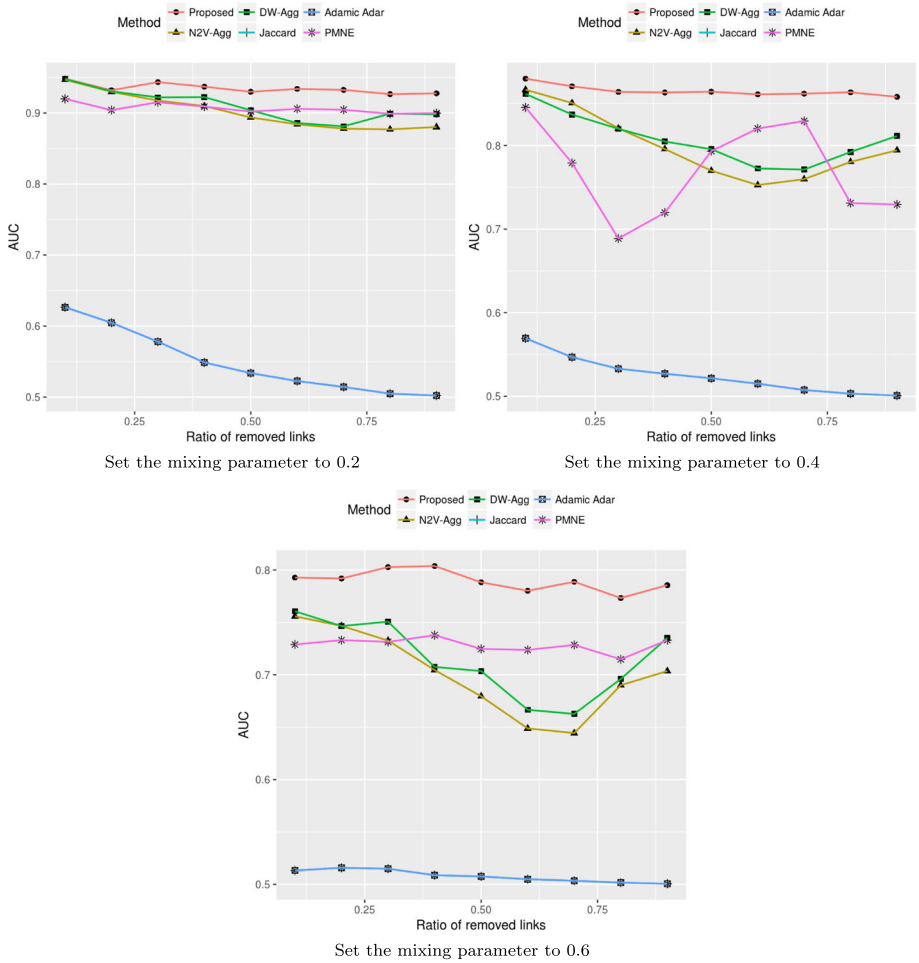

Fig. 4 Performance of the proposed model on different synthetic networks

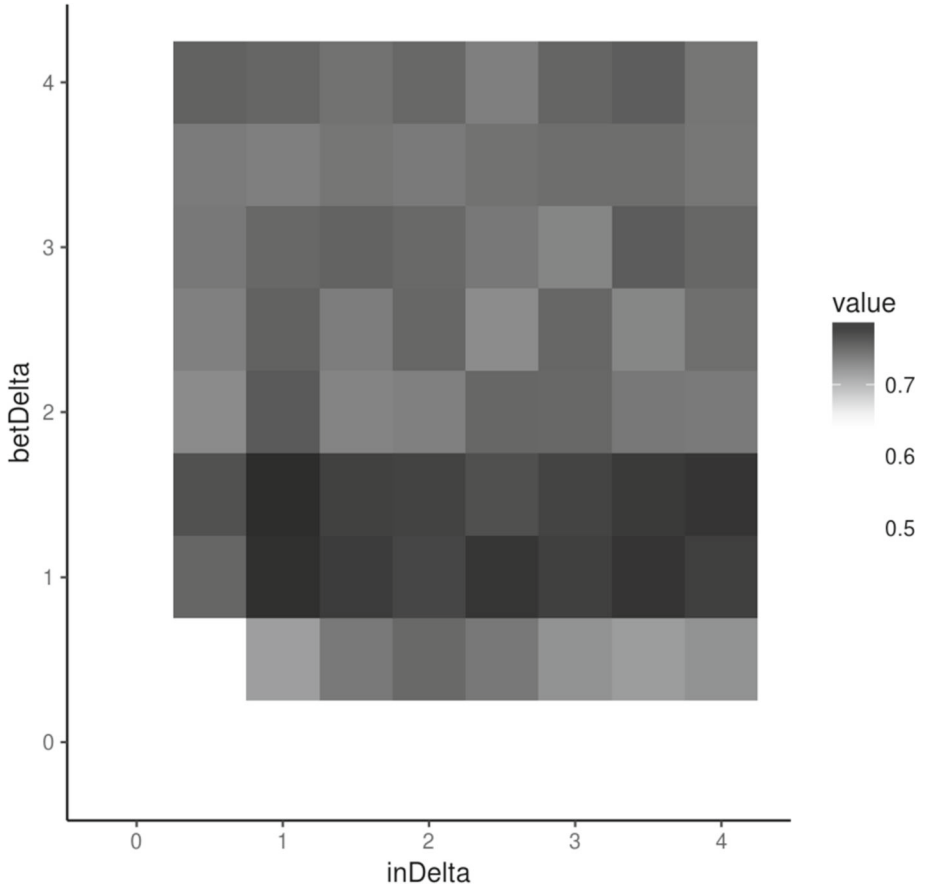

Fig. 5 The AUC values of the proposed model with different δ_{bet} and δ_{in} value on Celegans network

Through our experiment we figured it out that the range of 1 to 2 is a safe choice for both parameters.

Conclusion and future work

In this work, we propose MUNEM, a novel approach for learning a low-dimensional representation of a multiplex network using a triplet loss objective function. In our approach, we preserve the global structure of each layer, while at the same time fusing knowledge among different layers during the learning process. We evaluate the effectiveness of our proposed method by testing and comparing on both real-world and synthetic network. For the next step, we plan to design an embedding algorithm to find low-dimensional representations of dynamic networks. With some similarities with multiplex network, temporal network is more challenging network in both case of representation and learning.

Abbreviations

AUC: Area under the curve; MUNEM: Multiplex network embedding

Acknowledgements

Not applicable.

Authors' contributions

Author SH has been the lead author of the manuscript, who implemented the method and conducted the experiments. YL suggested the initial idea of the method and useful provide hints to generate synthetic multiplex network. YL and KP

has made contributions to design experiments and selecting the appropriate parameters of the method. All authors read and approved the final manuscript.

Funding
This work was partially supported by the NSF grant #1739413 and ARO Young Investigator Award W911NF-15-1-0599 (67192-NS-YIP).

Availability of data and materials
The datasets Celegans, Genetic, ArXiv and Climate are included in these published articles (Chen et al. 2006),(De Domenico et al. 2015), (De Domenico et al. 2015) and (Omodei et al. 2015), respectively. The synthetic multiplex generated in our study is available from the corresponding author on reasonable request.

Competing interests
The authors declare that they have no competing interests.

Received: 27 March 2019 Accepted: 28 November 2019
Published online: 23 December 2019

References

Aleta A, Moreno Y (2018) Multilayer networks in a nutshell. arXiv preprint arXiv:1804.03488
Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in Neural Information Processing Systems. pp 585–591
Chang S, Han W, Tang J, Qi G-J, Aggarwal CC, Huang TS (2015) Heterogeneous network embedding via deep architectures. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York. pp 119–128. https://doi.org/10.1145/2783258.2783296
Chen BL, Hall DH, Chklovskii DB (2006) Wiring optimization can relate neuronal structure and function. Proc Nat Acad Sci 103(12):4723–4728
De Domenico M, Lancichinetti A, Arenas A, Rosvall M (2015) Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems. Phys Rev X 5(1):011027
De Domenico M, Nicosia V, Arenas A, Latora V (2015) Structural reducibility of multilayer networks. Nature Commun 6:6864
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. ACM, New York. pp 855–864. http://doi.acm.org/10.1145/2939672.2939754. https://doi.org/0.1145/2939672.2939754
Javadi SHS, Gharani P, Khadivi S (2018) Detecting community structure in dynamic social networks using the concept of leadership. In: Sustainable Interdependent Networks: From Theory to Application. Springer, Cham. pp 97–118. https://link.springer.com/chapter/10.1007/978-3-319-74412-4_7
Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110
Le TM, Lauw HW (2014) Probabilistic latent document network embedding. In: 2014 IEEE International Conference on Data Mining. IEEE. pp 270–279. https://doi.org/10.1109/ICDM.2014.119. https://ieeexplore.ieee.org/abstract/document/7023344
Leskovec J, Lang KJ, Dasgupta A, Mahoney MW (2009) Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Int Math 6(1):29–123
Liu W, Chen P-Y, Yeung S, Suzumura T, Chen L (2017a) Principled multilayer network embedding. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE. pp 134–141. https://doi.org/10.1109/ICDMW.2017.23
Liu W, Wen Y, Yu Z, Li M, Raj B, Song L (2017b) Sphereface: Deep hypersphere embedding for face recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp 212–220. http://openaccess.thecvf.com/content_cvpr_2017/html/Liu_SphereFace_Deep_Hypersphere_CVPR_2017_paper.html
Ma Y, Ren Z, Jiang Z, Tang J, Yin D (2018) Multi-dimensional network embedding with hierarchical structure. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM, New York. pp 387–395. https://dl.acm.org/citation.cfm?id=3159680. https://doi.org/10.1145/3159652.3159680
Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems. pp 3111–3119
Omodei E, De Domenico MD, Arenas A (2015) Characterizing interactions in online social networks during exceptional events. Front Phys 3:59
Ou M, Cui P, Pei J, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York. pp 1105–1114. http://doi.acm.org/10.1145/2939672.2939751. https://doi.org/10.1145/2939672.2939751
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. https://dl.acm.org/citation.cfm?id=2623732
Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp 815–823
Stark C, Breitkreutz B-J, Reguly T, Boucher L, Breitkreutz A, Tyers M (2006) Biogrid: a general repository for interaction datasets. Nucleic Acids Res 34(suppl_1):535–539
Strano E, Shai S, Dobson S, Barthelemy M (2015) Multiplex networks in metropolitan areas: generic features and local effects. J Royal Soc Int 12(111):20150651
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, Republic and Canton of Geneva. pp 1067–1077. https://doi.org/10.1145/2736277.2741093. https://doi.org/10.1145/2736277.2741093
Tenenbaum JB, De Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323
Valente TW, Coronges K, Lakon C, Costenbader E (2008) How correlated are network centrality measures? Connections (Toronto, Ont.) 28(1):16
Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017a) Community preserving network embedding. In: Thirty-First AAAI Conference on Artificial Intelligence. https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14589
Wang S, Tang J, Aggarwal C, Chang Y, Liu H (2017b) Attributed Signed Network Embedding. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM, New York. pp 327–335. https://doi.org/10.1145/3132847.3132905
Zhang H, Qiu L, Yi L, Song Y (2018) Scalable multiplex network embedding. In: IJCAI. pp 3082–3088
Zitnik M, Leskovec J (2017) Predicting multicellular function through multi-layer tissue networks. Bioinformatics 33(14):190–198

Publisher's Note