Network Embedding via Coupled Kernelized Multi-dimensional Array Factorization

Linchuan Xu, Jiannong Cao, *Fellow, IEEE,* Xiaokai Wei, Philip S. Yu, *Fellow, IEEE*

Abstract—Network embedding has been widely employed in networked data mining applications as it can learn low-dimensional and dense node representations from the high-dimensional and sparse network structure. While most existing network embedding methods only model the proximity between two nodes regardless of the order of the proximity, this paper proposes to explicitly model multi-node proximities which can be widely observed in practice, e.g., multiple researchers co-author a paper, and multiple genes co-express a protein. Explicitly modeling multi-node proximities is important because some two-node interactions may not come into existence without a third node. By proving that LINE(1st), a recent network embedding method, is equivalent to kernelized matrix factorization, this paper proposes coupled kernelized multi-dimensional array factorization (Cetera) which jointly factorizes multiple multi-dimensional arrays by enforcing a consensus representation for each node. In this way, node representations can be more comprehensive and effective, which is demonstrated on three real-world networks through link prediction and multi-label classification.

Index Terms—Network embedding, kernelized array factorization, link prediction, multi-label classification

1 Introduction

NETWORK is a natural and ubiquitous data structure in various domains, such as social networks, gene co-expression networks, and co-authorship networks, simply because objects are related, and dependent on each other. As a result, many data mining applications involve network analysis, such link prediction [1], community detection [2], and node classification [3]. However, the usually high-dimensional and sparse structure of networks hampers the capabilities of data mining and machine learning models. More specifically, high dimensionality makes it inefficient to train data mining models while sparsity makes it difficult to generalize trained models on unseen data. Hence, network embedding [4] [5] [6] has been employed to learn low-dimensional and dense node representations that encode the network structure, which is basically achieved by presenting nodes close in the network representation to be close in a particular Euclidean space of interest.

Many network embedding methods have been proposed recently. Deepwalk [4] observes that sequences of nodes obtained by truncated random walks on the network representation can be regarded as sequences of natural language words, and then encodes node relationships through establishing the probabilities between a node in the sequence and other nodes (i.e., the context for generating the node of interest), which is achieved by employing Skip-gram [7], a word embedding model. node2vec [6] improves Deepwalk by designing a biased random algorithm which can explore diverse neighborhoods of a given node to obtain the node sequences. GraRep [8] proves that Skip-gram is equivalent to factorize the transition matrices of various orders of the network, and then proposes to separately factorize multiple transition matrices using singular value decomposition. LINE(1st) and LINE(2nd) [5] directly enforce nodes connected by the first-order links and by the second-order links to be close in the embedding space, respectively.

Almost all of them only explicitly model the proximity between two nodes regardless of the order of the proximity. Deepwalk [4] and node2vec [6] model the first-order and higher order proximities where the order depends on how close the two nodes are in a truncated random walk. GraRep [8] models the first-order up to a pre-defined kth-order proximity. LINE(1st) [5] and LINE(2nd) [5] model the first-order proximity and the second-order proximity, respectively.

However, in some scenarios, there are many interactions explicitly involving multiple nodes, e.g.,

- In gene co-expression networks [9] where interactions are established among genes when they co-express a protein, there exist certain proteins resulting from the co-expression of multiple genes,
- In academic social networks where interactions are established among researchers when they co-author a paper, there exist many papers which result from the collaborations of multiple researchers,
- In social networks where interactions are established among users participating in the same social circles, there exist social circles consisting of multiple users.

In the network representation, a multi-node interaction is reflected on all pairwise edges, e.g., each pair of all the researchers that co-authored a paper has an interaction in the co-authorship network. Therefore, a multi-node interaction can be regarded as a clique of a network since a clique is defined as a group of nodes such that each pair of nodes is connected. Moreover, a clique essentially represents the strongest structural cohesion among a group of nodes

- *Linchuan Xu and Jiannong Cao are with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong.*
  E-mail: {cslcxu, csjcao}@comp.polyu.edu.hk
- *Xiaokai Wei is with Facebook Inc., 1 Hacker Way, Menlo Park, CA, USA.*
  E-mail: weixiaokai@gmail.com
- *Philip S. Yu is with Department of Computer Science, University of Illinois at Chicago, IL, 60601, USA.*
  E-mail: psyu@uic.edu

[10]. Hence, a clique can be a general form of multi-node interaction. With the concept of a clique, we can also obtain multi-node interactions even though the given network does not explicitly define multi-node interactions, such as Facebook and Twitter friendship networks.

Multi-node interactions can be split into multiple two-nodes interactions, but presenting two nodes with interactions which only exist with a third node to be close in the embedding space without considering the third node is not appropriate. Moreover, there are some properties of multi-node interactions that can only be preserved when multi-node interactions are explicitly modeled. These two points are further explained as follows:

- Some two-node interactions may not even exist without other nodes, e.g., a particular paper with three authors may not be finished when one of the authors is missing, a particular protein co-expressed by multiple genes may not be produced without certain genes, and friendships induced by transitivity, i.e., friends of my friends are my friends, may not exist without the intermediate friends.
- The strengths of a two-node interaction with and without other nodes are different, e.g., the strengths of the friendship induced by transitivity may be different when the intermediate friends are in the presence and not in the presence, respectively.
- A two-node interaction may be shared by multiple multi-node interactions as illustrated in Fig. 1. The three three-node interactions may belong to different categories, e.g., three papers in three different domains, and three social circles with three different commonalities, such as hometown, university and interests. If solely looking at the two-node interactions between node 1 and node 2, the diversity in terms of categories or commonalities may be lost.

In this paper, we thus propose to explicitly model multi-node interactions in the context of network embedding. Assume that the example network in Fig. 1 defines a clique as a multi-node interaction, i.e., there are seven two-node interactions and three three-node interactions. To embed the network, the seven two-node interactions should be encoded into node representations as all existing methods do. The three-node interactions are also encoded into node representations as suggested above.

However, it is not trivial to extend from modeling two-node proximities to modeling multi-node proximities. For the popular Skip-gram based models like Deepwalk [4] and node2vec [6], two-node interactions are modeled through the input and output of a specially-designed neural network. It is challenging to design a neural network to take two nodes as inputs or assign two nodes to the same output in order to model three-node interactions. For methods modeling transition probabilities between nodes like GraRep [8], two-node interactions are modeled through the probabilities of a transition from one node to another node. It is challenging to extend to consider the transition path while modeling multi-node interactions.

Fortunately, we prove that LINE(1st) [5] is equivalent to kernelized matrix factorization where the matrix is the adjacency matrix and the kernel function is sigmoid function.

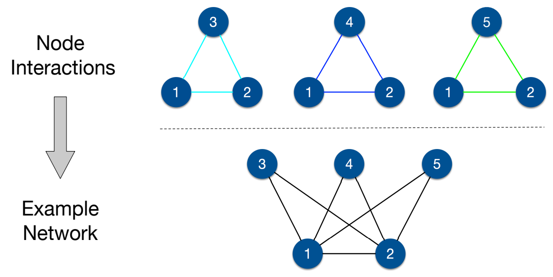


Fig. 1. An example network in which a two-node interaction between node 1 and node 2 shared by three three-node interactions.

More details about the proof can be found in Section 3. Motivated by the kernelized matrix factorization, we thus model multi-node interactions using multi-dimensional adjacency arrays, and then learn node representations via kernelized array factorization.

For a particular network, there may exist multiple arrays with different dimensions, which depends on the number of nodes in interactions. We assign a representation for each array, i.e., array-specific representation, to each node since the characteristics that nodes expose in the interactions of different number of nodes have different semantic meanings as suggested in Fig. 1. Moreover, multiple arrays are jointly factorized by enforcing a consensus representation for each node on which all array-specific representations agree. In this way, the consensus representation summarizes all array-specific characteristics and can be used to improve array-specific presentations in return. Hence, the proposed model is named as coupled kernelized multi-dimensional array factorization (Cetera).

Contributions of the paper are summarized as follows:

- To the best of our knowledge, this paper is the first one to explicitly model multi-node proximities in the context of network embedding.
- We prove that LINE(1st) is equivalent to kernelized matrix factorization where sigmoid function is employed as the kernel function.
- We propose Cetera to jointly factorize multiple multi-dimensional arrays via enforcing a consensus representation for each node.
- We demonstrate the effectiveness of Cetera on three real-world networks in applications including link prediction and multi-label classification.
- We present a comprehensive review of network embedding and propose a novel taxonomy for categorizing existing network embedding methods.

The rest of the paper is organized as follows. Section 2 presents preliminaries. In Section 3, we give the proof of the equivalence of LINE(1st) and kernelized matrix factorization. In Section 4, we develop the proposed Cetera model. Section 5 suggests the optimization algorithm. In Section 6, we provide empirical evaluation of Cetera. Section 7 presents the related work. In Section 8, we conclude this paper and introduce our future directions.

2 Preliminaries

In this paper, the studied network is defined as follows:

Definition 1 *$G(V, E)$ is a network, V is a set of nodes, and E is a set of directed or undirected, weighted or unweighted edges.*

In practice, $G(V, E)$ may have or not have explicit multi-node interactions. In this paper, the type of networks with explicit multi-node interactions is referred to as cooperation networks whose nodes co-operate in an event, such as co-authorship networks and gene co-expression networks. All the other networks are categorized as the second type of networks, such as friendship networks. Correspondingly, we name the second type as non-cooperation networks.

The way to define multi-node interactions for the two types of networks is different. For cooperation networks, multi-node interactions are only established among those with cooperations. Moreover, interactions of a larger number of nodes are propagated to interactions of smaller numbers of nodes, e.g., one four-node interaction produces four three-node and six two-node interactions, which is reasonable because the existence of a four-node cooperation must indicate the existence of a three-node cooperation. For non-cooperation networks, multi-node interactions are established among nodes forming cliques. The weight of each multi-node interaction is chosen as the smallest weight among all the two-node interactions.

In this paper, multi-node interactions are modeled by multi-dimensional arrays. An n-dimensional adjacency array for n-node interactions is defined as follows:

Definition 2 *An n-dimensional adjacency array is denoted as $\boldsymbol{A}^{(n)} \in \mathbb{R}^{|V|\times|V|\times|V|\times\cdots(\text{the number of } |V| \text{ is } n)}$. $\Omega^{(n)}$ denotes the index space. For each element $A^{(n)}_{i_1\ldots i_n}$=*

$$\begin{cases} weight & \text{if a interaction among node } i_1, \ldots, \text{and } i_n \\ 0 & otherwise, \end{cases} \tag{1}$$

where weight=1 for unweighted networks, i_1, ..., i_n are indices of nodes in each dimension, and $i_1 \neq \ldots \neq i_n$.

The two-dimensional array is also referred to as an adjacency matrix, and arrays of larger dimensions can also be referred to as tensors. For undirected networks, arrays of all dimensions are symmetric. For directed networks, the arrays are symmetric as well since for directed networks, multiple nodes form a clique if and only if there are two edges of different directions between each pair of nodes.

3 LINE(1st) as Light Kernelized Matrix Factorization

In this section, we prove that LINE(1st) [5] is equivalent to kernelized matrix factorization where the kernel is a similarity function operating in a high-dimensional and implicit feature space, and is popularly known in the filed of support vector machine (SVM). Note this proof is different from that of equivalence of LINE(1st) and matrix factorization without a kernel [11] [12]. In particular, the previous proof focused on finding a special matrix such that the dot product of the node embeddings can directly approximate the corresponding element of that matrix.

In LINE(1st), the proximity of two nodes is quantified as follows:

$$p(\boldsymbol{v}_1, \boldsymbol{v}_2) = \frac{1}{1 + \exp(-\boldsymbol{v}_1^T \boldsymbol{v}_2)}, \tag{2}$$

where $\boldsymbol{v}_1 \in \mathbb{R}^D$ and $\boldsymbol{v}_2 \in \mathbb{R}^D$ are representations of node 1 and 2, respectively, and D is the dimension of representations. $p(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is essentially a sigmoid kernel function, and is replaced by $\sigma(\boldsymbol{v}_1 \cdot \boldsymbol{v}_2)$ in the rest of the section.

Since the sigmoid function estimates the probability of linkage between two nodes, LINE(1st) presents each pair of nodes connected by an edge through minimizing KL divergence between the empirical probability distribution of linkage and the estimated distribution. After simple transformations, for unweighted networks, the loss function can be quantified as follows:

$$\mathcal{L}_{LINE(1st)} = -\sum_{(i,j)\in E} log\, \sigma(\boldsymbol{v}_i \cdot \boldsymbol{v}_j) \tag{3}$$

To avoid the trivial solution that $\forall i$ and $\forall d$, $\boldsymbol{v}_i^{(d)} = \infty$, i.e., the d_{th} dimension of node i is equal to infinity, LINE(1st) employs negative sampling to randomly sample multiple negative edges, i.e., non-existing edges, and presents each pair of nodes connected by a negative edge to be apart. As a result, the final loss function after ignoring the sampling process can be quantified as follows:

$$\mathcal{L}_{LINE(1st)} = -\sum_{(i,j)\in E} log\, \sigma(\boldsymbol{v}_i, \boldsymbol{v}_j) - \sum_{(h,k)\notin E} log\, \sigma(-\boldsymbol{v}_h \cdot \boldsymbol{v}_k), \tag{4}$$

which essentially applies cross entropy loss on each edge.

Eq. (4) thus can be reformulated by unifying $(i, j) \in E$ and $(h, k) \notin E$ into $(i, j) \in \Omega^{(2)}$ as follows:

$$\mathcal{L}_{LINE(1st)} = \sum_{(i,j)\in\Omega^{(2)}} \ell(A^{(2)}_{ij}, \sigma(\boldsymbol{v}_i \cdot \boldsymbol{v}_j)), \tag{5}$$

which is kernelized matrix factorization by employing sigmoid function as the kernel, and cross entropy loss as the loss function. Note that LINE(1st) is a light method because it employs negative sampling to reduce zero elements of the matrix in order to reduce computation costs.

4 The Proposed Cetera Model

Motivated by the equivalence of LINE(1st) and kernelized matrix factorization, Cetera learns node representations by performing kernelized array factorization by employing sigmoid function as the kernel function.

Similar to GraRep [8] which treats different kth-order proximities differently, i.e., learns a representation for each node from each order of proximity, Cetera assigns a representation for each node to preserve each multi-node proximity. Since multi-node proximities are modeled by multi-dimension arrays, each type of representation for each node is referred to as an array-specific representation. The motivation for learning array-specific representations is that different multi-node proximities involve a unique number of nodes, and have different semantic meanings as illustrated in Fig. 1.

But unlike GraRep learning order-specific representations separately, Cetera learns all array-specific representations simultaneously by enforcing consensus representations on which all array-specific representations agree. The consensus representation is motivated by the consensus principle utilized in multi-view learning [13] [14] because we can regard different arrays as different views of the network structure. In this way, the consensus representations can fuse information of all multi-node proximities. In return, consensus representations can improve each type of array-specific representations. The improved array-specific representations can achieve better array-specific tasks, such the link prediction between two nodes.

4.1 Array-specific Representation Learning

Because the learning of each array-specific representation is similar, we thus only take the learning from three-node proximities as an example. The loss function for factorizing a three-dimensional array can be quantified by following the kernelized matrix factorization, i.e., Eq. (5), as follows:

$$\mathcal{L}^{(3)} = \sum_{(i,j,k)\in\Omega^{(3)}} \ell(A^{(3)}_{ijk}, k^{(3)}(\boldsymbol{v}^{(3)}_i, \boldsymbol{v}^{(3)}_j, \boldsymbol{v}^{(3)}_k)), \tag{6}$$

where $k^{(3)}(\cdot)$ is a kernel function for quantifying three-node proximities defined as follows:

$$k(\boldsymbol{v}^{(3)}_i, \boldsymbol{v}^{(3)}_j, \boldsymbol{v}^{(3)}_k) = \frac{1}{1+\exp\{-\sum_d^D v^{(3,d)}_i v^{(3,d)}_j v^{(3,d)}_k\}}, \tag{7}$$

where $\boldsymbol{v}^{(3)}_i$ is representation for node i with respect to three-node proximities, and $v^{(3,d)}_i$ is the value of dimension d. $v^{(3,d)}_j$ and $v^{(3,d)}_k$ are similarly defined. Eq. (7) is an extension of Eq. (2) from two vectors to three vectors in terms of the sum of element-wise multiplication. Although the kernel function $k^{(3)}(\cdot)$ is not formally defined in the literature, it is well suitable for the purpose of this paper, i.e., to make the three embeddings to agree with each other, because the sum of element-wise multiplication can be reduced to the dot product of each vector and the combination of the other two vectors. Moreover, the experiments have demonstrated the effectiveness.

Cross entropy loss function $\ell(\cdot)$ can only be applied to unweighted networks. For weighted networks, we modify it as follows: $\ell(A^{(3)}_{ijk}, k(\boldsymbol{v}^{(3)}_i, \boldsymbol{v}^{(3)}_j, \boldsymbol{v}^{(3)}_k)) =$

$$\begin{cases} -A^{(3)}_{ijk} log\, k(\boldsymbol{v}^{(3)}_i, \boldsymbol{v}^{(3)}_j, \boldsymbol{v}^{(3)}_j) & A^{(3)}_{ijk} > 0 \\ -log(1 - k(\boldsymbol{v}^{(3)}_i, \boldsymbol{v}^{(3)}_j, \boldsymbol{v}^{(3)}_k)) & A^{(3)}_{ijk} = 0 \end{cases} \tag{8}$$

The modification is performed to reflect the edge weight which may indicate relationship strength since relationships with frequent interactions may be stronger than those with only one interaction.

4.2 Consensus Representation Learning

As mentioned before, we employ the consensus principle of multi-view learning [13] [14] to learn the consensus representations. The specific form of the consensus principle employed is centroid-based co-regularization [15], which is achieved by regularizing array-specific representations and the consensus representation to be similar to each other. Formally, the co-regularization is formulated as follows:

$$\min_{\boldsymbol{Z},\sum_n \boldsymbol{V}^{(n)}} \sum_n ||\boldsymbol{V}^{(n)} - \boldsymbol{Z}||^2_F, \tag{9}$$

where $\boldsymbol{Z} \in \mathbb{R}^{|V|\times D}$ and $\boldsymbol{V}^{(n)} \in \mathbb{R}^{|V|\times D}$ consist of consensus representations and node representations learned from n-node proximities, respectively, and $||\cdot||^2_F$ is Frobenius norm.

Algorithm 1 Alternating Optimization Algorithm

```
1:  Input: $G(V,E)$, D, $\lambda$, and $r$
2:  Output: $\boldsymbol{V}^{(1)}$, ..., $\boldsymbol{V}^{(n)}$, ..., $\boldsymbol{V}^{(N)}$, $\boldsymbol{Z}$

3:  Prepare multi-dimensional arrays
4:  Pre-train $\boldsymbol{V}^{(1)}$, ..., $\boldsymbol{V}^{(N)}$
5:  repeat
6:    Find the optimal $\boldsymbol{Z}$
7:    foreach $\boldsymbol{V}^{(n)}$ do
8:      Solve $\boldsymbol{V}^{(n)}$ by gradient descent with all the other
        variables fixed
9:  until $iteration = iteration_{max}$ or converge
10: end
```

Since this is the first work to consider multi-node interactions, Eq. (9) assumes all the representations are in the same Euclidean space. Different spaces and more sophisticated ways for the consensus learning are left as future studies.

Minimizing Eq. (9) w.r.t. $\boldsymbol{Z}$ thus can learn the consensus representations. In return, improving array-specific representations by utilizing consensus representations can be achieved by minimizing Eq. (9) w.r.t. array-specific representations, i.e., minimizing the following loss function:

$$\mathcal{L}_{\boldsymbol{V}^{(n)}} = ||\boldsymbol{V}^{(n)} - \boldsymbol{Z}||^2_F. \tag{10}$$

4.3 Joint Learning

By jointly considering all multi-node proximities, the final loss function for Cetera to embed the network structure can be quantified by direct addition as follows:

$$\begin{aligned}\mathcal{L}(\boldsymbol{V}^{(1)}, ..., \boldsymbol{V}^{(n)}, ...) =& \mathcal{L}^{(2)} + \mathcal{L}^{(3)} + \mathcal{L}^{(4)} ... + \lambda \sum_n ||\boldsymbol{V}^{(n)}||^2_F \\ &+ \sum_n ||\boldsymbol{V}^{(n)} - \boldsymbol{Z}||^2_F + \lambda||\boldsymbol{Z}||^2_F,\end{aligned} \tag{11}$$

where $\mathcal{L}^{(3)}$ is defined in Eq. (6) for preserving three-node proximities, $\mathcal{L}^{(2)}$ and $\mathcal{L}^{(4)}$ are similarly defined, $||\boldsymbol{V}^{(n)}||^2_F$ and $||\boldsymbol{Z}||^2_F$ are regularization terms for avoiding trivial solutions where the values of embeddings become very large, and $\lambda \in \mathbb{R}$ is the regularization coefficient. The model can include proximities involving any number of nodes. In the experiments, the model only includes a pre-defined set of proximities. The problem of determining the optimal number of nodes in the interactions is left as a future study.

5 The Optimization

The loss function in Eq. (11) is not jointly convex over all the variables, i.e., array-specific representations and consensus representations. We thus replace it with a sequence of easier sub-optimizations using an alternating algorithm [16] where each sub-optimization only solves one variable.

5.1 The Optimization Algorithm

Pseudo-codes of the optimization algorithm are presented in Algorithm 1. The input $r \in \mathbb{R}$ is negative ratio, which is the ratio of the number of positive multi-node edges to that of negative multi-node edges. The multi-dimensional array preparation has been presented in Section 3.

Before jointly learning all the variables, Algorithm 1 pre-trains each type of array-specific representations individually. 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 efficient and effective [17]. In our case, the objective function in Eq. (11) is not convex, and all array-specific representations are solved by gradient descent. Hence the initizalization largely determines the learning performance. The pre-training of each type of array-specific representations is performed by factorizing the corresponding array, which can also be solved by gradient descent, e.g., to pre-train $\boldsymbol{V}^{(3)}$ by only solving Eq. (6). For learning rate, we employ backtracking line search [18] to determine an appropriate one which can guarantee the descent in each iteration during the training process. Note that the optimization w.r.t. $\boldsymbol{Z}$ is convex, and the optimal solution can be obtained by straightforward linear algebra.

5.2 Complexity Analysis

The computation of Algorithm 1 is mainly spent on the multi-dimensional array preparation and the array factorization. For the cooperation networks, the time is used to propagate interactions of larger number of nodes into interactions of smaller number of nodes, which is essentially a combination problem. But in the experiments, we only include up to seven-node interactions. For the co-authorship network studied in this paper, the average number of authors is about three [19]. And the number of interactions starts to decrease when the number of nodes is 4 in Table 1. Hence, the time for array preparation for the first type of networks may be safely omitted.

For non-cooperation networks, we need to find multi-node interactions first. In particular, we obtain interactions of larger number of nodes by adding nodes to interactions of smaller number of nodes. Hence, the complexity of preparing up to seven-node interactions is $O(|V|(nnz(A^{(2)})/2! + nnz(A^{(3)})/3! + nnz(A^{(4)})/4! + nnz(A^{(5)})/5! + nnz(A^{(6)})/6! + nnz(A^{(7)})/7!))$ because arrays are symmetric, where $nnz(A^{(2)})$ is the number of non-zero elements of the two-dimensional array, and others are similarly defined. Since networks are usually sparse, the scalability to large-scale networks can be guaranteed.

With respect to array factorization, since we employ negative ratio to reduce the number of zero elements of the arrays, the complexity for each array is $O((r+1)Dnnz(A^{(n)}))$, where r is the negative ratio, D is the dimension of node representations, and n is the dimension of the array.

5.3 Convergence

Algorithm 1 is essentially a block-wise coordinate descent algorithm [20] with all the array-specific representations and the consensus representations as block variables. So convergence can be guaranteed based on the general proof of convergence for block-wise coordinate descent. Moreover, we observe Algorithm 1 converges very fast in terms of the outer iterations in the experiments as presented in the evaluation section.

TABLE 1
Network Statistics

Network	DBLP	Youtube	Flickr
# nodes	5091	8916	4700
# 2-node edges	17867	33802	224576
# 3-node edges	14956	45031	239147
# 4-node edges	7684	51270	373022
# 5-node edges	2931	41470	345385
# 6-node edges	859	21712	203930
# 7-node edges	172	6862	79847
# 8-node edges	–	1235	20965
# 9-node edges	–	128	3546
# 10-node edges	–	8	331
# labels	7	5	7
# labels per node	2.46	2.03	5.41
Type	undirected weighted	undirected unweighted	directed unweighted

6 Empirical Evaluation

6.1 Baselines

Cetera is evaluated against six recent network embedding methods, which are Deepwalk [4], LINE [5], GraRep [8], node2vec [6], NetMF [12], and PTE [21]. The first five models are designed for homogeneous networks while PTE is designed for heterogeneous networks. PTE is employed as a baseline because Cetera utilizes information about multi-node interactions from the first type of networks, such as paper-author information, which can be used to construct a paper-author bipartite network modeled in PTE [21]. But because Cetera does not utilize further information, such as label information, only the author-author bipartite network and the author-paper bipartite network are fed to PTE.

6.2 Datasets

Three real-word networks are studied as follows :

- DBLP [22]: In the experiments, we select several popular conferences from seven research fields which are SIGMOD, VLDB, ICDE, PODS, EDBT from Database, KDD, ICDM, SDM, PAKDD from Data Mining, AAAI, IJCAI, ICML, ECML from Machine Learning, and SIGIR, WWW, CIKM, WSDM, and ECIR from Information Retrieval, CVPR, ICCV, ECCV, ICIP, BMVC, WSCG, and ACCV from Computer Vision, ACL, EMNLP, COLING, NAACL, EACL, and CoNLL from Natural Language Processing, ICMR, ICME, ACMMM, and SIGMM from Multimedia. From these conferences, we select papers published during the time span from 2000 to 2009 with more than 2 authors, and select authors with at least two publications of different fields.
- Youtube [23]: This dataset contains friendship relationships. We select users that hold the group membership of any one of five major groups, which are 23, 30, 81, 82 and 367 indicated in the the dataset. We make the directed network undirected to increase the number of multi-node edges.
- Flickr [23]: This dataset contains friendship relationships among the users of Flickr. We select users that hold the group membership of any five of seven major groups, which are 135, 156, 172, 228, 295, 471 and 1098 indicated in the dataset.

TABLE 2
AUC (standard deviation) scores for DBLP future co-authorship prediction, where all the scores have been multiplied by 100% here and in the rest of the paper.

Method	Deepwalk	LINE(1st)	LINE(2nd)	PTE	GraRep	node2vec	NetMF (T=1)	NetMF (T=10)	**Cetera**
AUC	73.77(1.01)	60.65(0.89)	65.92(1.00)	70.12(1.12)	73.31(0.23)	72.22(1.15)	56.84(0.16)	68.24(0.21)	**76.16(0.97)**

TABLE 3
AUC(standard deviation) scores for DBLP multi-node co-authorship prediction, where W. A. is the abbreviation of "weighted average" and the weight is the number of test co-authorships.

AUC	Deepwalk	LINE(1st)	LINE(2nd)	PTE	GraRep	node2vec	NetMF (T=1)	NetMF (T=10)	**Cetera**
Three-node	52.85(1.32)	46.83(1.08)	57.18(1.12)	60.31(1.24)	27.70(0.93)	51.69(1.21)	47.77(0.96)	36.18(1.10)	**74.51(0.98)**
Four-node	79.74(1.51)	61.86(1.89)	71.94(1.43)	66.22(1.53)	**86.99(1.18)**	75.24(1.65)	69.55(0.16)	77.62(1.01)	65.03(1.29)
Five-node	52.72(1.19)	52.70(1.38)	48.17(1.28)	53.15(1.26)	13.47(1.16)	53.51(1.09)	46.74(1.16)	35.03(1.42)	**63.09(1.27)**
Six-node	76.45(1.31)	59.44(1.73)	62.98(1.32)	66.62(1.10)	**88.03(1.08)**	62.84(1.25)	57.26(1.36)	64.40(1.32)	68.30(1.22)
Seven-node	55.19(1.23)	53.63(1.28)	27.45(1.35)	48.12(1.25)	8.57(1.28)	58.21(1.18)	50.35(1.51)	44.18(1.61)	**71.84(1.18)**
W. A.	67.01	56.14	62.69	65.32	57.83	65.03	55.29	57.69	**73.33**

DBLP co-authorship network is the cooperation network while the Youtube and Flickr friendship networks belong to the non-cooperation network defined in Section 3. The number of two-node edges up to seven-node edges of the DBLP co-authorship network and two-node edges up to ten-node edges of the Youtube and Flickr friendship networks are summarized in Table 1. The research fields are used as the labels for DBLP, and the groups are used as labels for Youtube and Flickr.

6.3 Experiment settings

For the implementation, the dimension of node representations is set as the commonly used 128, the negative ratio is set as 5 as used in LINE, two-node proximities up to seven-node proximities are employed in the representation learning to make fair comparison with GraRep as GraRep models first-order proximities up to sixth-order proximities in its paper, regularization coefficient is set as 1.0, and 8 outer iterations are used as the maximum iterations. Our codes are written in Java and run on an Intel Genuine Intel(R) CPU @2.60GHZ 2.60GHZ server with 64 GB RAM.

We evaluate the methods in two applications, link prediction and multi-label classification. The dimension of the final node representations of GraRep is 6×128 since GraRep concatenates all order-specific representations. For Cetera, only the representations corresponding to 2-dimensional array are used for link prediction because link prediction is performed between two nodes instead of multiple nodes. In multi-label classification, we concatenate all array-specific representations and consensus representations.

6.4 Link Prediction

Link prediction is to infer new interactions among network nodes, and is typically performed by measuring pair-wise similarities because interactions are more likely to occur between similar nodes [1]. In the experiments, the similarities are firstly computed by the dot product of two node representations and then normalized by sigmoid function. The commonly used AUC (area under the curve) scores are employed as the performance metric. For DBLP, we perform future co-authorship prediction where co-authorships arising during the time span from 2010 to 2013 are employed as the positive test links, and the same number of negative test links are randomly sampled for the evaluation purpose. This process of learning and prediction is repeated 10 times. The results of average performance are presented in Table 2.

Table 2 shows the proposed Cetera achieves the best performance. The superior performance of other baselines to LINE and NetMF(T=1) may suggest the advantage of considering proximities of higher orders. PTE underperforms other baselines because the link prediction is to predict the first-order links which are not preserved in PTE.

One may have a concern that we explicitly model multi-node interactions but only employ the node presentations mainly learned from two-node interactions to perform the link prediction. The explanation is that the links to be predicted are essentially two-node interactions. But we can utilize characteristics of multi-node interactions to refine the node representations learned from two-node interactions because only looking at two-node interactions without considering other nodes is not inappropriate as explained in the introduction, e.g., the cooperation between two authors may not happen without a third author. To make a fair comparison with GraRep, the performance obtained by node representations corresponding to the first-order transition matrix factorized by GraRep is 63.19.

Besides the traditional link prediction, i.e., the prediction of co-authorships between two authors, we also study the prediction of co-authorships among multiple authors, which widely exist in practice. Similarly, the prediction is performed by measuring the multi-node similarities. We report the AUC scores of 10 times of experiments in Table 3. It shows that Cetera outperforms baselines in most times. Although some baselines have superior performance in other times, they have extremely unstable performance on different tasks, e.g., GraRep scores 88.03 on the six-node link prediction while scoring 8.57 on the seven-node link prediction. Hence, we present weighted average performance of each method where the weight is the number of test links (12708 two-node links, 8140 three-node links, 3912 four-node links, 1704 five-node links, 690 six-node links, 220 seven-node links). The result shows that Cetera consistently outperforms all the baselines.

For the Youtube and Flickr networks, we perform missing link prediction where partial links are used as train-

TABLE 4
AUC scores on link prediction when different ratios of links are used in the training phase.

Network	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Youtube	Deepwalk	62.55	66.23	69.56	72.66	75.18	77.86	79.01	80.23	80.89
	LINE(1st)	69.20	73.32	76.38	80.69	83.34	85.21	86.67	87.68	88.12
	LINE(2nd)	68.01	71.80	75.12	79.27	81.64	84.01	85.61	86.66	86.95
	GraRep	60.22	65.40	68.79	70.17	73.14	75.61	77.12	78.63	79.35
	node2vec	69.32	73.12	74.79	79.17	80.84	83.44	84.81	85.28	86.05
	NetMF(T=1)	65.12	68.55	71.13	73.66	76.75	78.22	79.36	80.16	80.82
	NetMF(T=10)	60.15	63.63	66.36	68.76	70.38	72.36	74.11	75.13	75.89
	Cetera	**75.32**	**82.68**	**85.60**	**87.19**	**88.56**	**89.96**	**90.90**	**91.34**	**91.55**
Flickr	Deepwalk	62.12	66.86	69.85	71.55	73.01	74.00	74.51	74.82	75.14
	LINE(1st)	57.66	60.36	63.12	67.96	69.12	70.57	72.97	74.72	74.82
	LINE(2nd)	55.26	57.69	61.02	63.01	62.90	62.96	62.70	62.72	62.82
	GraRep	56.19	60.11	62.36	66.55	68.95	70.23	71.31	72.58	72.82
	node2vec	56.90	61.22	64.95	67.76	69.03	70.67	71.35	72.82	73.36
	NetMF(T=1)	55.66	58.26	61.63	64.11	66.32	68.87	70.37	71.32	72.02
	NetMF(T=10)	58.62	62.16	65.15	67.95	69.41	71.60	72.31	73.02	73.52
	Cetera	**70.32**	**75.37**	**78.51**	**80.02**	**82.73**	**83.73**	**85.15**	**87.09**	**88.69**

TABLE 5
Micro-F1 and Macro-F1 scores obtained by different methods for multi-label classification.

Macro-F1	Deepwalk	LINE(1st)	LINE(2nd)	PTE	**GraRep**	node2vec	NetMF(T=1)	NetMF(T=10)	**Cetera**
DBLP	76.8	74.7	67.2	73.6	81.0	78.2	72.6	80.1	**81.3**
Youtube	66.3	65.3	61.9	–	84.3	63.6	55.6	59.6	**86.6**
Flickr	33.8	35.5	33.7	–	34.8	34.3	33.7	34.6	**38.9**
Micro-F1	Deepwalk	LINE(1st)	LINE(2nd)	PTE	**GraRep**	node2vec	NetMF(T=1)	NetMF(T=10)	**Cetera**
DBLP	80.5	79.6	74.8	76.6	83.3	81.2	79.4	82.7	**83.5**
Youtube	69.0	68.8	63.4	–	85.2	65.5	70.0	72.1	**87.8**
Flickr	71.7	72.5	71.7	–	69.6	72.1	71.6	72.1	**72.6**

TABLE 6
Micro-F1 and Macro-F1 scores obtained by array-specific node representations learned by Cetera for multi-label classification.

Micro-F1	two-dimensional	**three-dimensional**	four-dimensional	**five-dimensional**	six-dimensional	seven-dimensional
DBLP	79.0	**81.5**	76.9	77.3	77.4	78.1
Youtube	86.5	86.6	86.7	**86.9**	86.8	86.5
Flickr	71.3	**71.6**	70.5	71.4	71.5	71.5
Macro-F1	two-dimensional	**three-dimensional**	four-dimensional	**five-dimensional**	six-dimensional	seven-dimensional
DBLP	70.8	**77.3**	67.5	67.5	67.8	69.0
Youtube	85.0	85.2	85.2	**85.5**	85.4	85.1
Flickr	33.3	**33.6**	31.7	33.4	33.5	33.5

ing data and the remaining ones are used as test links. Specifically, we perform nine runs of experiments where the training links range from 10% to 90% of the whole links. The results are reported in Table 4, which shows similar patterns to the results for the DBLP co-authorship prediction. We can see there exist many cliques as shown in Table 1, e.g., the number of 3-node cliques is even larger than that of two-node edges. The cliques provide extra information for learning representations from two-node edges, and hence Cetera obtains the best performance.

6.5 Multi-label Classification

For DBLP, each researcher may publish papers in more than one research field, and for Youtube and Flickr, each user may belong to more than one group. Hence, in multi-label classification, more than one label are assigned to each data point. We employ binary-relevance SVM with polynomial kernel implemented in Meka [24] as the classifier, use 5-fold cross validation as the evaluation method, and report Micro-F1 and Macro-F1 scores in Table 5.

It shows that Cetera obtains the best performance on all the datasets. The advantage of Cetera over baselines can be explained mainly by two reasons. Firstly, two-node proximities are essentially partial information of multi-node proximities because two-node interactions are split from multi-node interactions for the cooperation networks, or build up multi-node interactions for those non-cooperation networks. Hence, Cetera can actually capture more information than all the baselines. One may note that the improvement on the DBLP dataset over baselines is not as remarkable as on the Youtube and the Flickr datasets. Note that for the DBLP dataset, the number of multi-node interactions is less than that of two-node interactions as presented in Table 1. Therefore, the benefits brought by modeling multi-node interactions may be limited.

To demonstrate the effectiveness of multi-node interactions, we present the performance obtained by array-specific node representations in Table 6. It is interesting to note that node representations learned from two-node interactions never obtain the best performance, which can be explained by the second reason presented below.

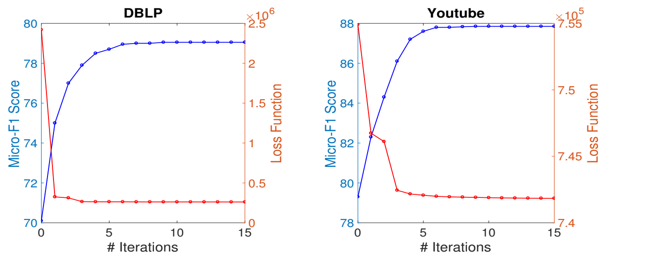


Fig. 2. The performance of multi-label classification with respect to the number of iterations.

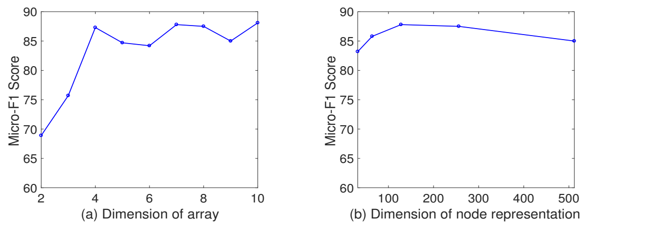


Fig. 3. The performance of multi-label classification with respect to the dimension of arrays (a) and the dimension of representations (b).

Secondly, each data point usually has more than one label, e.g., the average number of labels per node is more than 2 as indicated in Table 1. The diversity of label information can be preserved more effectively by explicitly modeled multi-node interactions as illustrated in Fig. 1, i.e., a two-node interaction may be shared by multiple three-node interactions in different domains (i.e., labels).

It is worth noting that the advantage of Cetera and GraRep over other baselines is not brought by dimension advantage. Actually, a larger dimension usually does not bring better performance as studied in this paper and in baseline themselves, such as in LINE and GraRep.

6.6 Convergence Analysis

In this section, we study the convergence of Algorithm 1. Specifically, we study the performance of the algorithm on applications with respect to the number of outer iterations where each iteration takes about 719 seconds and 1158 seconds on average for the DBLP and Youtube network, respectively. We only present the experiments on multi-label classification for the DBLP and Youtube network in Fig. 2 because other experiments show similar results. Fig. 2 shows that the algorithm converges fast and can usually converge to stable performance after about 5 iterations.

6.7 Parameter Sensitivity

This section evaluates how Algorithm 1 is sensitive to the dimension of arrays and the dimension of representations. In the experiments, we set the largest dimension of arrays from 2 to 10, and dimensions of representations from the choices of $\{32, 64, 128, 256, 512\}$. The performance on the multi-label classification measured by Micro-F1 scores for the Youtube network is presented in Fig. 3. Fig. 3 (a) shows Algorithm 1 can achieve stable performance after including 4-node edges. Fig. 3 (b) shows Algorithm 1 is not much sensitive to the dimension of node representations if the dimension is not too small (e.g., 32) or too large (e.g., 512).

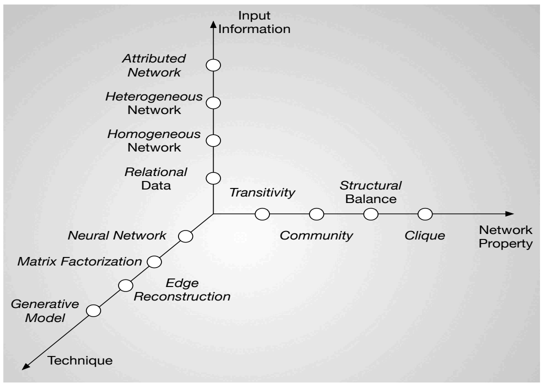


Fig. 4. Three-dimensional network embedding taxonomy space.

7 Related Work

In this section, we first present a comprehensive review of existing network embedding methods, and then present the relevance of the proposed Cetera to existing methods.

Learning low-dimensional data representations by embedding the network structure has been studied since early 2000s, and was referred to as graph embedding [25], [26], [27], [28], [29] then. But graph embedding is actually designed for data points which are initially independent, i.e., no relationships or interactions among them. Hence, before performing the embedding, previous methods need to establish links between data points where links are usually based on neighbor relationships, e.g., k-nearest neighbors or neighbors within a pre-defined radius of distance. In the rest of this section, graph embedding is also referred to as early network embedding.

Instead, recent network embedding is designed for data points with natural linkage relationships. Not needing to make artificial links, recent network embedding methods can utilize additional information, explore various network properties, and have diverse techniques. In this paper, we thus categorize existing network embedding methods according to three criteria, i.e., the input information, the explored network property, and the developed technique.

We present the summary of the taxonomies in a three-dimensional space corresponding to the aforementioned three criteria as illustrated in Fig. 4. On each dimension, we only list the main-stream elements. For instance, on the dimension of input information, almost all the embedding methods are designed for one of the four inputs, i.e., relational data, homogeneous networks, heterogeneous networks, and attributed networks. With respect to the comparison with existing taxonomies, our taxonomies apply the classification criteria parallelly instead of a tree structure [30] in which different classification criteria are adopted and applied sequentially. We prefer a parallel classification because the three criteria are at the same levels. Moreover, our taxonomies have different classification criteria from and one more criterion than the taxonomies in [31].

7.1 Classification by Input Information

A network is a homogeneous network when it only has one type of nodes and one type of edges while a heterogeneous network has more than one type of nodes or more than one type of edges. A network is an attributed network when its nodes or edges have attributes, e.g., a co-authorship network can be an attributed network when the authors have paper keywords as attributes.

7.1.1 Relational Data as Input

Almost all methods for relational data belong to early network embedding. Network embedding can be applied to relational data because it is assumed that real-world data presented in high-dimensional spaces are expected to concentrate in the vicinity of a manifold of much lower dimensionality [33]. Hence, network embedding is essentially to model the structure of the data-supporting manifold, which explains why early network embedding is also referred to as manifold learning [33].

Most early network embedding methods employ edge reconstruction to learn node representations, i.e., minimizing the distance of nearby data points in the manifold [34] or maximizing the probability of generating existing edges using node representations [29]. The methods based on minimzing distance include [25], [26], [27], [28], [35] while the methods based on maximizing probabilities include [29].

7.1.2 Homogeneous Networks as Input

The recent network embedding for homogeneous networks starts with Deepwalk [4] which is based on Skip-gram [7]. Another popular model is node2vec [6] which extends Deepwalk to consider communities or roles that nodes belong to because nodes belonging to the same community and roles should be close in the embedding space.

There are also many other embedding methods that are not based on Skip-gram, such as LINE [5], GraRep [8], HOPE [36], and M-NMF [37]. Most of them except for LINE(1st) [5] preserve network properties while embedding the network structure, and hence are introduced in more details in Section 7.3, i.e., the section of classification by network property.

7.1.3 Heterogeneous Networks as Input

With the success of Skip-gram based models on homogeneous networks, there are also many Skip-gram based heterogeneous network embedding models, such as metapath2vec [38], HIN2Vec [39], and HINE [40]. The commonality of these methods is to design random walk algorithms for walking on heterogeneous networks, which generates sequences of heterogeneous nodes, or meta-paths.

Besides, PTE [21] embeds three bipartite networks where the three networks share a set of nodes, e.g., word-word network, word-document network, and word-label network studied in its paper. PTE employs LINE(2nd) [5] to embed each bipartite network. HNE [41] embeds heterogeneous information networks with two types of nodes and three types of edges. More specifically, the two types of nodes are image and text, and the three types of edges are image-image edges, image-text edges, and text-text edges. The edges are established when nodes have similar semantic meanings or have citation relationships. MVE [42] regards each type of edges as one view of the given heterogeneous network. MVE adopts a method similar to LINE [5] to embed each view of the network, and then jointly embeds multiple views via an attention mechanism.

7.1.4 Attributed Networks as Input

Attributes provide additional information about nodes and can make node representations more comprehensive.

TADW [43], shorted for Text-associated Deepwalk, embeds both the network structure and node attributes extracted from text-based content through matrix factorization. AANE [44] learns node representations by not only minimizing the difference of pairs of nodes that are connected but also minimizing the difference between the similarity computed by node representations and the similarity computed by node attributes.

Node label can also be viewed as node attributes. Hence, we classify semi-supervised network embedding methods into this category. Both MMDP [45] and DDRW [46] jointly learn node representations in the way of Deepwalk and train a SVM classifier on the learned node representations. Similarly, [47] also jointly learns node representations and trains a classifier. Moreover, [47] proposes an inductive learning method which estimates a parametric function for nodes not seen during the training phase

7.2 Classification by Technique

Neural network for network embedding is usually feed-forward artificial neural networks which are specially designed for network embedding. Matrix factorization is the commonly used method for decomposing a matrix into product of matrices which usually consist of node representations in the context of network embedding. Edge reconstruction is more like a principle that the distance of a pair of nodes that are connected should be considerably small in the embedding space of interest. Generative models for network embedding emphasize that the observed edges of a particular node are generated by the underlying conditional distribution, and learn node representations by maximizing the likelihood of edges in the network.

7.2.1 Neural Network

Skip-gram based models, such as Deepwalk [4], node2vec [6], struc2vec [48], and even metapath2vec [38], employ neural network models for network embedding, because Skip-gram [7] is essentially specially-designed neural network model. Different from traditional neural network models, the input to the input layer is a vector of one-hot representation of nodes, the weights of the input layer to the first hidden layer is essentially node representations, and the output estimates the probabilities of interactions of the input node to all the other nodes.

There are also other kinds of neural network models used in network embedding. HNE [41] employs deep neural network models to capture the interactions between heterogeneous components, i.e., CNN for image nodes and FC layers for text nodes. ProjE [49] designs a special neural network with a combination layer and a projection layer. Moreover, it defines a point-wise loss (similar to multi-class classification) and a list-wise loss (i.e., softmax regression loss) for knowledge graph embedding.

7.2.2 Matrix Factorization

As mentioned in Section 7.1.4, TADW [43] is essentially a matrix factorization model, i.e., decomposing a transition matrix into product of matrices. Similarly, GraRep [8] factorizes transition matrices with different transition steps using singular value decomposition (SVD). A recent study [12] proposes a unified matrix factorization model and proves that Deepwalk, node2vec, LINE and PTE can all be put under the unified model.

7.2.3 Edge Reconstruction

Almost all early network embedding models are based on edge reconstruction, i.e., minimizing the distance of pairs of nodes connected by edges or maximizing the probability of generating existing edges using node representations as mentioned in Section 7.1.1. There are also recent network embedding models based on minimizing distance, such as AANE [44] and HNE [41], and maximizing probabilities, such as LINE(1st) [5].

7.2.4 Generative Model

Network embedding methods based on generative models emphasize that the observed edges of a particular node are generated by the underlying conditional distribution, and these models then learn node representations by maximizing the likelihood of edges in the network. In this sense, Deepwalk and node2vec mentioned above can also be regarded as generative models because they use random walk algorithms to sample "context" nodes for each node, and attempt to maximize the likelihood of observing "context" nodes for the given node.

Recently, the game-theoretical min-max framework proposed in generative adversarial nets (GAN) [50] to estimate generative models has been widely adopted in various domains including network embedding [51] [52]. The basic idea of GAN is to jointly estimate two models, i.e., a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample comes from the training data rather than G.

7.3 Classification by Network Property

Transitivity means that links are transitive from nodes to nodes, e.g., friends of my friends are my friends. By considering transitivity, potential links and high-order links can be incorporated in node representations. Community is a group of nodes which have more intra-group links than inter-group links, which widely exist in real-world networks. Structural balance [53] [54] considers the possible ways in which triangles on three individuals can be signed, and posits that triangles with three positive signs and those with one positive sign are more plausible than triangles with two positive signs or none. Hence, structural balance is only studied in signed network embedding. A clique is a group of nodes such that each pair of nodes are connected.

7.3.1 Transitivity

There are many methods [5], [8], [36], [55] exploring the transitivity property, which essentially extends the first-order neighbor relationships to higher-order relationships.

LINE(2nd) [5] explores the principle that the degree of overlap of two people's friendship networks correlates with the strength of ties between them [56]. Essentially, LINE(2nd) preserves second-order links.

As mentioned in Section 7.2.2, GraRep [8] learns node representations by factorizing transition matrices. Actually, GraRep factorizes first-order up to a pre-defined k-th order transition matrices because links are transitive. Based on the success of utilizing high-order proximities among nodes, NEU [57] summarizes that high-order proximities are beneficial to learn node representations, and proposes a general method to incorporate high-order proximities into existing methods, e.g., Deepwalk, node2vec and GraRep.

All the methods above assume the transition property is symmetric, but HOPE [36] observes that the transitivity is asymmetric in directed networks, and proposes an embedding method especially for directed networks.

7.3.2 Community

Two nodes in the same community usually have more commonalities than two nodes in different communities. Then it is meaningful to encode community information into node representations.

To achieve this purpose, both M-NMF [37] and ComE [58] jointly perform network embedding and community detection. More specifically, on the one hand, they learn node representations by embedding the network structure. On the other hand, they learn the community membership of each node. The joint learning is achieved by approximating community structures using the multiplication of node representations and community representations.

7.3.3 Structural Balance

All the network embedding methods mentioned above are designed for networks with only positive edges. But in practice, there are negative relationships in some scenarios, such as dis-like and dis-trust relationships. Hence, some networks may be signed networks with both positive and negative edges, such as Slashdot and Epinion [54]. Methods mentioned above are not suitable for embedding signed networks since negative links have different semantic meanings from positive links. Moreover, it is less optimal to separately embed the positive sub-network and the negative sub-network. Hence, some methods explore the structural balance theory, such as SiNE [59] and SNEA [60]

7.4 The Relevance of Cetera to Existing Methods

According to the three-dimensional taxonomy space in Fig. 4, the proposed Cetera takes homogeneous networks as input, uses matrix factorization technique, and preserves the strongest structural cohesion, i.e., clique, for those networks which do not explicitly define multi-node interactions.

The major difference from existing methods in terms of the dimension of input information is that Cetera explicitly models multi-node interactions as mentioned in the introduction. The major difference in terms of technique is that Cetera performs kernelized array factorization in stead of matrix factorization without a kernel. Moreover, Cetera assigns a consensus representation to each node in order to jointly factorize multiple arrays. The major difference in terms of network property is that Cetera explores the strongest structural cohesion for the first time.

8 Conclusion and Future Work

This paper proposes Cetera to explicitly model multi-node proximities using multi-dimensional adjacency arrays in network embedding for the first time. The proposed Cetera learns node representations by factorizing these arrays by employing sigmoid function as the kernel function. Moreover, motivated by the consensus principle of multi-view learning, Cetera enforces consensus learning by fusing all array-specific representations into consensus representations, which are expected to improve each type of array-specific representations in return. Through the evaluation on three real-world networks in link prediction and multi-label classification, we demonstrate the advantage of Cetera over six recent models. In the future, we plan to study how to determine the optimal n in n-node proximities.

Acknowledgments

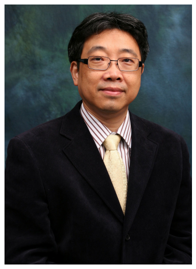
The work described in this paper was partially supported by National Key R&D Program of China, Project No: 2018YFB1004801, Hong Kong RGC Collaborative Research Fund (CRF), Project No. C5026-18G, Hong Kong RGC Collaborative Research Fund (CRF), Project No. C6030-18G, NSF through grants III-1526499, III-1763325, III-1909323, SaTC1930941, and CNS-1626432.

References

[1] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *journal of the Association for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.
[2] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3, pp. 75–174, 2010.
[3] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," in *Social network data analytics*. Springer, 2011, pp. 115–148.
[4] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.
[5] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.
[6] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.
[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[8] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 891–900.
[9] J. M. Stuart, E. Segal, D. Koller, and S. K. Kim, "A gene-coexpression network for global discovery of conserved genetic modules," *science*, vol. 302, no. 5643, pp. 249–255, 2003.
[10] J. Moody and D. R. White, "Structural cohesion and embeddedness: A hierarchical concept of social groups," *American Sociological Review*, pp. 103–127, 2003.
[11] Q. Wang, Z. Wang, and X. Ye, "Equivalence between line and matrix factorization," *arXiv preprint arXiv:1707.05926*, 2017.
[12] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifyingdeepwalk, line, pte, and node2vec," *arXiv preprint arXiv:1710.02971*, 2017.
[13] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv preprint arXiv:1304.5634*, 2013.
[14] J. Zhao, X. Xie, X. Xu, and S. Sun, "Multi-view learning overview: Recent progress and new challenges," *Information Fusion*, vol. 38, pp. 43–54, 2017.
[15] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Advances in neural information processing systems*, 2011, pp. 1413–1421.
[16] J. C. Bezdek and R. J. Hathaway, "Some notes on alternating optimization," in *AFSS International Conference on Fuzzy Systems*. Springer, 2002, pp. 288–300.
[17] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle *et al.*, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, p. 153, 2007.
[18] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific Journal of mathematics*, vol. 16, no. 1, pp. 1–3, 1966.
[19] M. E. Newman, "The structure of scientific collaboration networks," *Proceedings of the National Academy of Sciences*, vol. 98, no. 2, pp. 404–409, 2001.
[20] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, 2001.
[21] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1165–1174.
[22] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008.
[23] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM, 2007, pp. 29–42.
[24] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes, "Meka: a multi-label/multi-target extension to weka," *Journal of Machine Learning Research*, vol. 17, no. 21, pp. 1–5, 2016.
[25] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.
[26] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[27] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
[28] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in neural information processing systems*, 2002, pp. 585–591.
[29] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Advances in neural information processing systems*, 2003, pp. 857–864.
[30] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *arXiv preprint arXiv:1711.08752*, 2017.
[31] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques and applications," *arXiv preprint arXiv:1709.07604*, 2017.
[32] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 1, pp. 40–51, 2007.
[33] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.
[34] A. Talwalkar, S. Kumar, and H. Rowley, "Large-scale manifold learning," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.
[35] T. F. Cox and M. A. Cox, *Multidimensional scaling*. CRC press, 2000.
[36] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding." in *KDD*, 2016, pp. 1105–1114.
[37] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding." in *AAAI*, 2017, pp. 203–209.
[38] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017.

[39] T.-y. Fu, W.-C. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1797–1806.
[40] Z. Huang and N. Mamoulis, "Heterogeneous information network embedding for meta path based proximity," *arXiv preprint arXiv:1701.05291*, 2017.
[41] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 119–128.
[42] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1767–1776.
[43] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina*, 2015, pp. 2111–2117.
[44] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 633–641.
[45] C. Tu, W. Zhang, Z. Liu, and M. Sun, "Max-margin deepwalk: Discriminative learning of network representation." in *IJCAI*, 2016, pp. 3889–3895.
[46] J. Li, J. Zhu, and B. Zhang, "Discriminative deep random walk for network classification." in *ACL (1)*, 2016.
[47] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *arXiv preprint arXiv:1603.08861*, 2016.
[48] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 385–394.
[49] B. Shi and T. Weninger, "Proje: Embedding projection for knowledge graph completion." in *AAAI*, 2017, pp. 1236–1242.
[50] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
[51] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "Graphgan: Graph representation learning with generative adversarial nets," *arXiv preprint arXiv:1711.08267*, 2017.
[52] Q. Dai, Q. Li, J. Tang, and D. Wang, "Adversarial network embedding," *arXiv preprint arXiv:1711.07838*, 2017.
[53] F. Heider, "Attitudes and cognitive organization," *The Journal of psychology*, vol. 21, no. 1, pp. 107–112, 1946.
[54] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 2010, pp. 1361–1370.
[55] C. Yang, M. Sun, Z. Liu, and C. Tu, "Fast network embedding enhancement via high order proximity approximation."
[56] M. Granovetter, "The strength of weak ties.-american journal of sociology. vol. 78, is. 6. p. 1360-1380," 1973.
[57] C. Yang, M. Sun, Z. Liu, and C. Tu, "Fast network embedding enhancement via high order proximity approximation," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*, 2017, pp. 19–25.
[58] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 377–386.
[59] S. Wang, J. Tang, C. Aggarwal, Y. Chang, and H. Liu, "Signed network embedding in social media," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 327–335.
[60] S. Wang, C. Aggarwal, J. Tang, and H. Liu, "Attributed signed network embedding," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 137–146.

Linchuan Xu received the B.E. degree in information engineering from Beijing University of Posts and Telecommunications in 2013, and Ph.D. degree from Department of Computing of the Hong Kong Polytechnic University, in 2018. He is currently a post-doctoral researcher of Department of Mathematical Informatics, Graduate School of Information Science and Technology at the University of Tokyo. His current research interests include data mining and big data with emphasis on network data analytics and medical data analytics.



Jiannong Cao received the B.Sc. degree in computer science from Nanjing University, China, in 1982, and the M.Sc. and Ph.D. degrees in computer science from Washington State University, USA, in 1986 and 1990 respectively. He is currently a Chair Professor of Department of Computing at The Hong Kong Polytechnic University, Hong Kong. His research interests include parallel and distributed computing, wireless networks and mobile computing, big data and cloud computing, pervasive computing, and fault tolerant computing. He has co-authored 5 books in Mobile Computing and Wireless Sensor Networks, co-edited 9 books, and published over 500 papers in major international journals and conference proceedings. He is a fellow of IEEE, a distinguished member of ACM, a senior member of China Computer Federation (CCF).

Xiaokai Wei received the B.S. degree from Beijing University of Posts and Telecommunications and Ph.D. degree from University of Illinois at Chicago, both in computer science. He joined Facebook Inc. as a research scientist in 2016. His main research areas are data mining and machine learning, especially feature selection and social network mining. He has published more than 20 papers in refereed journals and conferences, such as WWW, WSDM, AAAI, AISTATS, ICDM, SDM, ECML/PKDD.

Philip S. Yu received the B.S. Degree in E.E. from National Taiwan University, the M.S. and Ph.D. degrees in E.E. from Stanford University, and the M.B.A. degree from New York University. He is a Distinguished Professor in Computer Science at the University of Illinois at Chicago and also holds the Wexler Chair in Information Technology. Before joining UIC, Dr. Yu was with IBM, where he was manager of the Software Tools and Techniques department at the Watson Research Center. His research interest is on big data, including data mining, data stream, database and privacy. He has published more than 1,000 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. Dr. Yu is a Fellow of the ACM and the IEEE. Dr. Yu is the recipient of ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on mining, fusion and anonymization of big data, the IEEE Computer Society?s 2013 Technical Achievement Award for "pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining and anonymization of big data", and the Research Contributions Award from IEEE Intl. Conference on Data Mining (ICDM) in 2003 for his pioneering contributions to thefield of data mining. He also received the ICDM 2013 10-year Highest-Impact Paper Award, and the EDBT Test of Time Award (2014). He was the Editor-in-Chiefs of ACM Transactions on Knowledge Discovery from Data (2011-2017) and IEEE Transactions on Knowledge and Data Engineering (2001-2004).