Multi-Level Network Embedding with Boosted Low-Rank Matrix Approximation

Jundong Li^{*}, Liang Wu[†], Ruocheng Guo[‡], Chenghao Liu[◊], Huan Liu[‡]^{*}University of Virginia, jl6qk@virginia.edu[†]Airbnb Inc., liang.wu@airbnb.com[‡]Arizona State University, {rguo12, huan.liu}@asu.edu[◊]Singapore Management University, chliu@smu.edu.sg

Abstract—As opposed to manual feature engineering which is tedious and difficult to scale, network embedding has attracted a surge of research interests as it automates the feature learning on graphs. The learned low-dimensional node vectors ease the knowledge discovery on graphs by enabling various off-the-shelf machine learning tools to be directly applied. Recent research has shown that the past decade of network embedding approaches either explicitly factorize a carefully designed matrix or are closely related to implicit matrix factorization, with the fundamental assumption that the factorized node connectivity matrix is low-rank. Nonetheless, the global low-rank assumption does not necessarily hold especially when the factorized matrix encodes complex node interactions, and the resultant single low-rank embedding matrix is insufficient to capture all the observed connectivity patterns. In this regard, we propose a novel multi-level network embedding framework BoostNE, which can learn multiple node embeddings of different granularity from coarse to fine without imposing the prevalent global low-rank assumption. The proposed BoostNE method is also in line with the successful gradient boosting method in ensemble learning. We demonstrate the superiority of the proposed BoostNE framework by comparing it with existing state-of-the-art network embedding methods on various datasets.

Index Terms—Network Embedding; Multi-Level; Low-Rank Matrix Approximation; Boosting; Residual Matrix

I. INTRODUCTION

Learning discriminative representations of nodes in a network is essential for various network analytical tasks as it avoids the laborious manual feature engineering process. Additionally, as the node embeddings are often learned in a task-agnostic fashion, they are generalizable to a number of downstream learning tasks [1]–[3]. On top of that, it also has broader impacts in advancing many real-world applications, ranging from recommendation [4], polypharmacy side effects prediction [5], to name disambiguation [6]. The basic idea is to represent each node by a low-dimensional vector in which the relativity information among nodes is maximally transcribed.

A vast majority of existing network embedding methods can be broadly divided into two different yet highly correlated cat-

egories. Firstly, early network embedding methods are largely based on the matrix factorization approaches. In particular, methods in this family represent the topological structure among nodes as a deterministic connectivity matrix and leverage low-rank matrix approximation strategies to obtain the node vector representations. Various methods differ in the way on how the connectivity matrix is built, and typical examples include modularity matrix [7], Laplacian matrix [8], k -step transition matrix [9], and higher-order adjacency matrix [10]. Secondly, the recent advances of network embedding research are largely influenced by the skip-gram model [11]. The key idea behind these algorithms is that nodes tend to have similar embedding representations if they co-occur frequently on short random walks over the network [1], [3] or are directly connected with each other within certain contexts [12], [13]. Even though the two families of network embedding methods are distinct in nature, recent studies [14] found their inherent correlations and then cast the skip-gram inspired network embedding methods in the matrix factorization framework.

To this end, in this work, we investigate the network embedding problem within the framework of matrix factorization due to its broad generalizability. As mentioned previously, after the closed-form connectivity matrix for each network embedding model is derived, the principled way to permit node embedding is to perform low-rank matrix approximation, such as with eigendecomposition, singular value decomposition (SVD), and nonnegative matrix factorization (NMF). Here, the fundamental assumption is that the closed-form connectivity matrix is low-rank (a.k.a. *global low-rank*), and the factorized matrix is sufficient to provide a “one-size fits all” representation to encode the connectivity information among nodes. However, the widely perceived assumption is untenable in practice and may further hinder us learn effective node embedding representations due to the following reasons. On one hand, the connectivity matrix in real-world scenarios is often not sparse and encodes diverse connectivity patterns among nodes, making the global low-rank assumption improper [15]. On the other hand, simply relying on the global low-rank property for a single representation for nodes cannot fully explain how the nodes are connected in the network. In a nutshell, it is desired to learn the embedding in a forward stagewise fashion to gradually shift the focus to the unexplained node connectivity behaviors as stages progress.

To address the above problems, we propose a multi-level

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ASONAM '19, August 27–30, 2019, Vancouver, Canada

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6868-1/19/08...\$15.00

http://dx.doi.org/10.1145/3341161.3342864

Notations	Definitions or Descriptions
$G = (\mathcal{V}, \mathcal{E})$	the given network
n	number of nodes in G
m	number of edges in G
d	dimension of the final embedding representation
d_s	dimension of each single level embedding
k	levels of the final multi-level embedding
v_i	the i -th node in G
$\mathbf{A} \in \mathbb{R}_+^{n \times n}$	adjacency matrix of G
$\mathbf{D} \in \mathbb{R}_+^{n \times n}$	diagonal node degree matrix of G
$\mathbf{S} = \mathbf{D}^{-1} \mathbf{A}$	transition probability matrix of G
$\mathbf{U}_j \in \mathbb{R}_+^{n \times d_s}$	embedding representation of the j -th level
$\mathbf{U} \in \mathbb{R}_+^{n \times kd_s}$	the final embedding representation
$\mathbf{d}$	degree vector of nodes
$\mathbf{S}^k$	k -step transition probability matrix
$\text{vol}(G)$	volume of G ($\text{vol}(G) = \sum_i \sum_j \mathbf{A}_{ij}$)

TABLE I: Symbols.

network embedding framework BoostNE to obtain multiple granularity views (from coarse to fine) of the network over the full spectrum. The proposed framework is motivated by the gradient boosting framework [16] in ensemble learning, which learns multiple weak learners sequentially and then combines them together to make the final prediction. The main contributions of this paper are summarized as follows:

- We examine the fundamental limitations of existing network embedding approaches, especially the methods that directly leverage or can be reduced to the framework of matrix factorization.
- We propose a novel network embedding framework BoostNE to identify multiple embeddings from coarse to fine in a forward stagewise manner.
- We conduct experiments on real-world networks from various domains to verify the superiority of the proposed BoostNE. Further studies are presented to understand why the proposed BoostNE works and how the number of levels impacts the final embedding representation.

II. PROBLEM DEFINITION

We denote matrices using bold uppercase characters (e.g., $\mathbf{X}$), vectors with bold lowercase characters (e.g., $\mathbf{x}$), scalars with normal lowercase characters (e.g., x). The i -th element of vector $\mathbf{x}$ is denoted by x_i . The i -th row, the j -th column and the (i, j) -th entry of matrix $\mathbf{X}$ are denoted by $\mathbf{X}_{i*}$, $\mathbf{X}_{*j}$, and $\mathbf{X}_{ij}$ respectively. The transpose of matrix $\mathbf{X}$ is $\mathbf{X}'$. The Frobenius norm of matrix $\mathbf{X}$ is denoted as $\|\mathbf{X}\|_F$. The r -th power of matrix $\mathbf{X}$ is denoted as $\mathbf{X}^r$. The main symbols used throughout this paper are summarized in Table I.

Let $G = (\mathcal{V}, \mathcal{E})$ be the given network, where $\mathcal{V} = \{v_1, \dots, v_n\}$ is the node set and $\mathcal{E} = \{e_1, \dots, e_m\}$ is the edge set. We use the matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ to denote the adjacency matrix of the network, where $\mathbf{A}_{ij} \geq 0$ is a real number denoting the edge weight between node v_i and node v_j . In this work, we focus on undirected network such that $\mathbf{A}_{ij} = \mathbf{A}_{ji}$ though our problem can be easily extended to directed networks. We formally define the problem of multi-level network embedding as follows.

Problem 1. (Multi-Level Network Embedding): Given a network $G = (\mathcal{V}, \mathcal{E})$ with n nodes, a predefined embedding dimensionality d and the number of levels k , we try to learn

a series of k embeddings $\mathbf{U}_i \in \mathbb{R}^{n \times d_s}$ ($i = 1, \dots, k$) for each node in the network ($d = kd_s \ll n$). The target is to ensure that the complex node connectivity information among nodes is gradually preserved from coarse to fine in a forward stagewise manner as the levels progress from 1 to k .

III. THE PROPOSED BOOSTNE FRAMEWORK

The key idea of the proposed BoostNE framework is to first formulate the network embedding problem within the framework of matrix factorization by constructing a closed-form node connectivity matrix, then it successively factorizes the residual (unexplained part) obtained from the previous stage. To this end, it enables us to generate a sequence of embedding representations of different granularity views from coarse to fine. Afterwards, we ensemble all these weak embedding representations as the final embedding for downstream learning tasks. An illustrative example of the workflow of the proposed BoostNE is shown in Fig. 1.

A. Network Embedding as Matrix Factorization

As mentioned previously, a large number of existing network embedding methods are fundamentally based on the techniques of matrix factorization. Concretely, they target at building a deterministic connectivity matrix of various forms to capture different types of connections among nodes, including the 1st-order proximity, the higher-order proximity, the community structure, and etc. The other category of methods is mostly inspired by the language model word2vec [11]. They either exploit random walks on the network to capture the node structural relations or directly perform edge modeling to learn the structure-preserving node representations. Even though their optimization target is distinct from matrix factorization based methods, recent studies [14], [17] examined and found that most of these algorithms can also be reduced to the matrix factorization framework, and the desired embedding representations can be derived by performing principled matrix factorization methods. Next, we briefly show the connections between these sophisticated network embedding methods and matrix factorization.

Spectral Clustering [8]: Given the adjacency matrix $\mathbf{A}$ and the diagonal degree matrix $\mathbf{D}$, the node embedding representation $\mathbf{U}$ can be obtained by concatenating the top- d eigenvectors of the normalized adjacency matrix $\mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$.

Graph Factorization [18]: The node embedding $\mathbf{U}$ is obtained by performing symmetric matrix factorization on the adjacency matrix $\mathbf{A}$ such that $\mathbf{U} = \arg\min_{\mathbf{U}} \|\mathbf{A} - \mathbf{U} \mathbf{U}'\|_F^2$.

Social Dimension [7]: It constructs the modularity matrix as $\mathbf{A} - \mathbf{d} \mathbf{d}' / 2m$, where $\mathbf{d}$ is the node degree vector. Then the node embedding $\mathbf{U}$ can be identified by the top- d eigenvectors of the modularity matrix.

GraRep [9]: It concatenates the top left singular vectors of a series of transition matrices $\mathbf{X}_p$ ($p = 1, \dots, K$). In particular, K denotes the number of transition steps, $\mathbf{X}_p = \max(\mathbf{Y}_p, 0)$ with $(\mathbf{Y}_p)_{ij} = \log(\mathbf{S}_{ij}^p / \sum_t \mathbf{S}_{tj}^p) - \log(b)$, where b is the log shift factor and is normally set as $1/n$.

Hope [19]: It derives the connectivity matrix as $\mathbf{M} = \mathbf{M}_g^{-1} \mathbf{M}_l$ where high-order proximity matrices $\mathbf{M}_g$ and $\mathbf{M}_l$ are built

Fig. 1: An illustration of the proposed framework - BoostNE. BoostNE first constructs the complex node connectivity matrix $\mathbf{X}$ from the adjacency matrix $\mathbf{A}$. Motivated by the principle of gradient boosting, the proposed BoostNE successively factorizes the residual of the connectivity matrix from the previous stage and obtains embedding representations from coarse (e.g., $\mathbf{U}_1$) to fine (e.g., $\mathbf{U}_3$). The final embedding representation $\mathbf{U}$ is obtained by ensembling all previous weak embedding representations.

from $\mathbf{A}$. Then it performs low-rank matrix factorization on $\mathbf{M}$ to get the node embeddings.

Deepwalk [1]: Deepwalk makes an analogy between truncated random walks on networks with sentences in a text corpus and learns the node embedding as the skip-gram model. It has been shown [14] that Deepwalk is equivalent to factorize a transformation of network’s normalized Laplacian matrix $\log(\text{vol}(G)(\frac{1}{T} \sum_{r=1}^T \mathbf{S}^r) \mathbf{D}^{-1}) - \log(b)$, where $\text{vol}(G)$ denotes the volume of the network G , T denotes the window size, and b is the number of negative samples.

node2vec [3]: It factorizes a matrix related to the stationary distribution and transition probability of 2nd-order random walk. It is shown in [14] that the factorized matrix is $\log(\frac{\frac{1}{2T} \sum_{r=1}^T (\sum_u \mathbf{Y}_{wu} \mathbf{P}_{cu}^r + \sum_u \mathbf{Y}_{cu} \mathbf{P}_{wu}^r)}{(\sum_u \mathbf{Y}_{wu})(\sum_u \mathbf{Y}_{cu})}) - \log(b)$, where $\mathbf{P}_{uvw}$ is the random walk probability to node u given the current visited node v and previously visited node w . $\mathbf{Y}$ encodes the 2nd-order random walk stationary distribution.

LINE [12]: It is a special case of Deepwalk when the size of context is specified as one [14] and it is equivalent to factorize the matrix $\log(\text{vol}(G) \mathbf{S} \mathbf{D}^{-1}) - \log(b)$.

In this following context, we focus on the node connectivity matrix of Deepwalk with the following reasons: (1) it is more general than other node connectivity matrices as it captures both local and global interactions among nodes in the network; (2) it often leads better learning performance as shown in [14].

B. Multi-Level Network Embedding with Boosted Low-Rank Matrix Approximation

As per the above summarization, after various node connectivity matrices are derived from the original network topology, a vast majority of existing network embedding methods discover the node embeddings with the prevalent assumption that the underlying connectivity matrix is of low-rank. To this end, they target at learning the node embeddings with low-rank matrix approximation methods by conjecturing that the discriminative connectivity information should be well encoded within a single low-dimensional node representation.

As in the case of Deepwalk, the resultant node connectivity matrix is a nonnegative matrix, and to permit meaningful embedding representations, one possible solution is to perform

nonnegative matrix factorization (NMF) on the connectivity matrix $\mathbf{X} \in \mathbb{R}_+^{n \times n}$ with a global low-rank assumption:

$$\min_{\mathbf{U}, \mathbf{V} \geq 0} \|\mathbf{X} - \mathbf{UV}\|_F^2, \quad (1)$$

where d denotes the dimension of the embeddings. The factorized matrices $\mathbf{U} \in \mathbb{R}_+^{n \times d}$ and $\mathbf{V} \in \mathbb{R}_+^{d \times n}$ are both low-rank ($d \ll n$) and nonnegative. We can interpret $\mathbf{U}$ as the embedding of nodes that act like a “center” node while $\mathbf{V}$ can be regarded as the embedding of nodes that play the role of “context” node [20]. The above objective function can be solved by conventional optimization algorithms [21], [22].

In the above formulation, in order to perform NMF, the factorized matrix $\mathbf{X}$ has to satisfy the property of global low-rank. However, it is often argued that the low-rank assumption does not necessarily hold in real-world scenarios, especially when complex interactions among nodes are involved [15], [23]. In light of this, performing a single NMF on the connectivity matrix $\mathbf{X}$ may lead to suboptimal results, and the obtained “one-size fits all” embedding representation is insufficient to encode the connectivity patterns among nodes.

To find a better embedding representation that well approximates the node connectivity matrix $\mathbf{X}$, we relax the low-rank assumption of a single run of factorization. Instead, we assume that the connectivity matrix can be well approximated by performing multiple levels of NMF, resulting in the following objective function:

$$\min_{\mathbf{U}_i, \mathbf{V}_i \geq 0, i=1, \dots, k} \|\mathbf{X} - \sum_{i=1}^k \mathbf{U}_i \mathbf{V}_i\|_F^2, \quad (2)$$

where k denotes the number of levels, and $\mathbf{U}_i \in \mathbb{R}_+^{n \times d_s}$ and $\mathbf{V}_i \in \mathbb{R}_+^{d_s \times n}$ denotes the embedding representation of the center node and the context node in the i -th level during the multi-level factorization process.

In this work, inspired by the well-known ensemble learning methods [24], [25], we propose to learn multiple levels of node representations by using a forward stagewise strategy. The essential idea is to perform a sequence of NMF operations with each single NMF operation focusing on the residual of the previously not well-explained part. Hence, the

Algorithm 1 The proposed multi-level network embedding framework - BoostNE**Input:** A given network $G = (\mathcal{V}, \mathcal{E})$; number of levels k , embedding dimension in each level d_s .**Output:** The node embedding representation $\mathbf{U}_1, \dots, \mathbf{U}_k$ from coarse to fine and the final embedding representation $\mathbf{U}$.

- 1: Obtain the node connectivity matrix $\mathbf{X}$ from the adjacency matrix $\mathbf{A}$ of the network;
- 2: **for** $i = 1$ to k **do**
- 3: Compute the residual matrix $\mathbf{R}_i$ with Eq. (3);
- 4: Obtain the center node embedding representation $\mathbf{U}_i$ and the context node embedding representation $\mathbf{V}_i'$ by alternating optimization algorithms;
- 5: **end for**
- 6: Return the final embedding as $\mathbf{U} = [\mathbf{U}_1, \dots, \mathbf{U}_k]$

initial embedding representations provide a coarse view of the node connectivity patterns while the latter embeddings provide finer-grained embedding representations. In other words, different stages present various views of the embedding of different granularity. Specifically, in the i -th level, the not well explained residual matrix is defined as follows:

$$\mathbf{R}_i = \begin{cases} \mathbf{X} & \text{if } i = 1 \\ \max(\mathbf{R}_{i-1} - \mathbf{U}_{i-1}\mathbf{V}_{i-1}, 0) & \text{if } i \geq 2. \end{cases} \quad (3)$$

The max operation implies that if there exist any negative elements after the approximation, we convert it to be zero. With the above defined residual matrix, the embedding representation at the i -th level is obtained by minimizing the following loss function:

$$\min_{\mathbf{U}_i, \mathbf{V}_i \geq 0} \|\mathbf{R}_i - \mathbf{U}_i\mathbf{V}_i\|_F^2. \quad (4)$$

Compared with the objective function in Eq. (1) which returns the node embedding representation in a single run with NMF, our proposed algorithm returns multiple weak representations from coarse to fine in a greedy fashion. In other words, once the earlier level embeddings $\mathbf{U}_i$ and $\mathbf{V}_i'$ are obtained, they are fixed for the remaining operations. It should be noted that the similar idea of boosted low-rank matrix approximation has been used for the local topic discovery problem [25], where the document-by-term matrix is taken as an input. In BoostNE, we adapt the idea to perform a series of factorization operations on the network connectivity matrix and its residuals for embedding representation learning. Additionally, in [25], the local weighting scheme is used to deliver high-quality local topics to users, while we empirically find that the similar weighting scheme cannot guarantee better node embeddings. Next, we briefly analyze the time complexity of the proposed BoostNE and NMF with a single run in Eq. (1), here we specify $d = d_s k$ for a fair comparison. The time complexity of optimizing Eq. (1) is related to the number of nonzero elements in $\mathbf{X}$ and the rank d , which is $\#iterations \times O(\text{nnz}(\mathbf{X})d)$. While for the proposed BoostNE, as can be observed from the illustrative example in Fig. 1, the residual matrix (the unexplained part) becomes sparser and sparser as the level goes up, thus the computational cost of the proposed BoostNE is $\#iterations \times O(\text{nnz}(\mathbf{R}_1 + \dots + \mathbf{R}_k)d_s)$. As

$\text{nnz}(\mathbf{R}_1) + \dots + \text{nnz}(\mathbf{R}_k) < \text{nnz}(\mathbf{X})k$, the proposed BoostNE is more efficient than performing a single run NMF in obtaining a “one-size fits all” embedding representation. The pseudocode of the proposed BoostNE is shown in Algorithm 1.

IV. EXPERIMENTS

We conduct experiments to assess the effectiveness of the proposed BoostNE. Before introducing the experimental results, we first introduce the datasets, baseline methods, and experimental settings. In this section, we also perform further analysis on why the multi-level embedding method achieves better performance by studying the approximation error of the connectivity matrix. At last, we investigate how the number of levels k affects the final network embedding results.

A. Dataset

We collect and use four real-world network datasets from different domains for experimental evaluation. All these four datasets are publicly available and have been used in previous research [1], [3], [17]. The detailed descriptions of these datasets are as follow:

- **Cora**¹: Cora is a citation network of scientific publications. It consists of 2,708 papers from 7 different research areas. There is a total number of 5,278 citation links in the dataset.
- **Wiki**²: Wiki is a collection of wikipedia documents that are inherently connected with each other via hyperlinks. Each document is categorized into a number of predefined classes denoting their topics. In total, we have 2,363 documents, 11,596 hyperlinks, and 17 topics.
- **PPI**³: It is a subgraph of the protein-protein interaction network of Homo Sapiens. The labels of the protein are obtained from the hallmark gene sets and denote the biological states. We have 3,860 proteins, 37,845 interaction links, and 50 states.
- **Blogcatalog**⁴: It is a social blogging website in which users follow each other and post blogs under certain predefined categories. The main categories of blogs by the users are regarded as the class labels of users. In the dataset, we have 10,312 users, 333,983 user relations and 39 predefined categories.

For all the above-mentioned datasets, we have removed all self-loop edges and have converted bi-directional edges to undirected ones for a fair comparison of various network embedding methods.

B. Compared Baseline Methods

In this subsection, we compare our proposed multi-level network embedding framework BoostNE with existing efforts mentioned previously from two main categories: matrix factorization based network embedding methods and skip-gram inspired network embedding methods. Among them, Spectral Clustering [8], Social Dimension [7], GraRep [9] belong to

¹<https://linqs.soe.ucsc.edu/data>²<https://github.com/thunlp/OpenNE/tree/master/data/wiki>³<https://snap.stanford.edu/node2vec/#datasets>⁴<http://socialcomputing.asu.edu/datasets/BlogCatalog>

the former category while Deepwalk [1] and LINE [12] fall into the latter. In addition, we also compare with a recently proposed framework NetMF [14] which provides a general framework to factorize a closed-form node connectivity matrix for embedding learning. In this work, to have a fair comparison with our method, we adapt it by performing NMF instead of SVD after the closed-form connectivity matrix is derived.

For all the compared network embedding methods, we follow [1], [3], [14] to specify the final embedding dimension as 128. The baseline methods of GraRep, Deepwalk, and LINE are implemented in the OpenNE toolkit⁵, and we follow their default parameter settings. In particular, for GraRep, the transition step is specified as 4. For Deepwalk, we set the number of walks as 10, the walk length as 80, and the window size as 10. In terms of LINE, we concatenate both the 1st-order and the 2nd-order embedding representations, the negative sample size is 5, and the number of training epochs is 100. Lastly, for NetMF, we use the Deepwalk’s implicit matrix by following the same setting as Deepwalk, and the parameter h in NetMF is set as 256. For the proposed BoostNE, the only additional parameter we need to specify is the number of levels k , we set it as 8. More discussions on the impact of k will be presented later.

C. Experimental Settings

To assess the effectiveness of the proposed multi-level network embedding framework BoostNE, we follow the commonly adopted setting to compare different embedding algorithms on the task of multi-label node classification. In the task of multi-label node classification, each node is associated with multiple class labels, our target is to build a predictive learning model to predict the correct labels of nodes. In particular, after the embedding representations of all n nodes are obtained, we randomly sample a portion of nodes as the training data, use their embeddings and their class labels to build a predictive classification model, and then make the predictions with the embeddings of the remaining nodes. In the experiments, we repeat the process 10 times and report the average classification performance in terms of Micro-F1 and Macro-F1, which are widely used multi-label classification evaluation metrics. Normally, higher values imply better classification performance, which further indicates better node embedding representations. In the experiments, we vary the percentage of training data from 10% to 90%, and the logistic regression in Liblinear [26] is used as the discriminative classifier.

D. Embedding Results Comparison

We compare the quality of embeddings of different methods in the task of multi-label node classification. The comparison results are shown in Table II, Table III, Table IV, and Table V. We make the following observations from these tables:

- For all methods, the multi-label node classification performance w.r.t. Micro-F1 and Macro-F1 gradually increases when the portion of training data is varied from 10% to 90%. It implies that more training data can help us learn better models for node classification.

Fig. 2: Approximation error of the node connectivity of NetMF and BoostNE with different level number k .

- The proposed multi-level network embedding method BoostNE achieves the best classification performance in most of the cases. To further validate the conclusion, we perform a pairwise Wilcoxon signed rank test between BoostNE and other embedding methods and the results indicate that BoostNE is significantly better when the p -value threshold is set as 0.05.
- Compared with NetMF which learns node embedding within a single run of low-rank matrix approximation, the proposed BoostNE learns multiple embedding representations from coarse to fine by gradually factorizing the residual matrix from the previous stage. The improvement of BoostNE over NetMF shows the effectiveness of the ensemble methods as multiple weak embedding representations lead to a more effective embedding.
- Most of the time, the skip-gram inspired methods such as Deepwalk and LINE are better than conventional matrix factorization based methods. Even though both NetMF and BoostNE are kinds of matrix factorization based methods, their performance is much better than conventional matrix factorization methods as the connectivity matrix they factorize encode complex node interactions.

E. Further Analysis w.r.t. Approximation Error

Until now, we have shown that the proposed multi-level network embedding method is superior to NetMF with a single run of NMF on the node connectivity matrix. Despite its empirical success, the underlying reason why performing multiple stages of low-rank approximation leads to better embedding representations still remain opaque. In this subsection, we investigate the underlying mechanism from the approximation error perspective. In particular, we vary the number of levels k in the range of $\{2, 4, 8, 16, 32, 64\}$ and compare the Frobenius norm of the residual matrix after multiple stages of low-rank approximation. The baseline method NetMF can be regarded as a special case when $k = 1$. The comparison results on these four datasets are shown in Fig. 2.

As can be observed from the figure, the approximation of the node connectivity matrix from BoostNE is much smaller than that of NetMF. In particular, the approximation error gradually goes down when the value of k is increased from 2 to 64. This observation helps us explain why the proposed multi-level network embedding framework BoostNE learns better

⁵<https://github.com/thunlp/OpenNE>

Training Ratio		10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	Spectral Clustering	0.6814	0.7597	0.7777	0.7775	0.8091	0.7986	0.8058	0.8157	0.8170
	Social Dimension	0.5459	0.6604	0.7004	0.7252	0.7423	0.7605	0.7635	0.7832	0.7789
	GraRep	0.7588	0.7770	0.7844	0.7868	0.7897	0.7866	0.7925	0.7939	0.7952
	Deepwalk	0.7589	0.7874	0.7989	0.8039	0.8151	0.8181	0.8118	0.8175	0.8162
	LINE	0.6588	0.7126	0.7271	0.7439	0.7514	0.7598	0.7700	0.7655	0.7609
	NetMF	0.7379	0.7829	0.8002	0.8072	0.8120	0.8152	0.8180	0.8275	0.8306
	BoostNE	0.7824	0.8047	0.8178	0.8250	0.8257	0.8266	0.8314	0.8367	0.8373
Macro-F1	Spectral Clustering	0.6746	0.7552	0.7699	0.7764	0.8053	0.7928	0.8004	0.8104	0.8071
	Social Dimension	0.5032	0.6331	0.6860	0.7130	0.7247	0.7532	0.7540	0.7742	0.7705
	GraRep	0.7455	0.7657	0.7733	0.7768	0.7779	0.7742	0.7822	0.7802	0.7817
	Deepwalk	0.7457	0.7776	0.7886	0.7973	0.8074	0.8094	0.8047	0.8133	0.8049
	LINE	0.6409	0.7007	0.7158	0.7319	0.7423	0.7513	0.7599	0.7564	0.7545
	NetMF	0.7220	0.7706	0.7902	0.7977	0.8025	0.8063	0.8109	0.8229	0.8264
	BoostNE	0.7638	0.7906	0.8062	0.8142	0.8143	0.8171	0.8232	0.8313	0.8309

TABLE II: Node classification performance comparison on the Cora dataset.

Training Ratio		10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	Spectral Clustering	0.5333	0.5761	0.6236	0.6353	0.6460	0.6432	0.6597	0.6432	0.6419
	Social Dimension	0.3920	0.5054	0.5727	0.5991	0.6182	0.6289	0.6386	0.6282	0.6496
	GraRep	0.5962	0.6233	0.6360	0.6444	0.6461	0.6502	0.6531	0.6510	0.6616
	Deepwalk	0.5935	0.6324	0.6485	0.6606	0.6654	0.6865	0.6732	0.6907	0.6713
	LINE	0.5609	0.6032	0.6283	0.6356	0.6552	0.6659	0.6729	0.6732	0.6768
	NetMF	0.5849	0.6306	0.6508	0.6642	0.6709	0.6786	0.6817	0.6848	0.6916
	BoostNE	0.6113	0.6442	0.6625	0.6709	0.6749	0.6846	0.6829	0.6915	0.7013
Macro-F1	Spectral Clustering	0.4012	0.4459	0.4774	0.4959	0.5107	0.5111	0.5364	0.5119	0.4952
	Social Dimension	0.3154	0.4082	0.4584	0.4802	0.4946	0.5013	0.5036	0.5021	0.5028
	GraRep	0.4127	0.4383	0.4575	0.4690	0.4745	0.4785	0.4825	0.4829	0.4988
	Deepwalk	0.4368	0.4907	0.5051	0.5347	0.5353	0.5475	0.5531	0.5695	0.5492
	LINE	0.4314	0.4655	0.4917	0.5088	0.5448	0.5359	0.5409	0.5560	0.5334
	NetMF	0.4188	0.4595	0.4797	0.4998	0.5192	0.5304	0.5364	0.5391	0.5455
	BoostNE	0.4421	0.4742	0.5055	0.5248	0.5404	0.5496	0.5562	0.5645	0.5692

TABLE III: Node classification performance comparison on the Wiki dataset.

Training Ratio		10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	Spectral Clustering	0.1629	0.1881	0.1942	0.2027	0.2085	0.2108	0.2131	0.2177	0.2086
	Social Dimension	0.1250	0.1399	0.1622	0.1798	0.1795	0.1880	0.1945	0.1985	0.1966
	GraRep	0.1933	0.2118	0.2218	0.2291	0.2393	0.2418	0.2474	0.2458	0.2532
	Deepwalk	0.1588	0.1754	0.1889	0.1987	0.2075	0.2075	0.2155	0.2234	0.2238
	LINE	0.1342	0.1548	0.1706	0.1811	0.1923	0.2014	0.2063	0.2096	0.2180
	NetMF	0.1771	0.2033	0.2171	0.2264	0.2329	0.2401	0.2466	0.2531	0.2565
	BoostNE	0.1883	0.2137	0.2271	0.2367	0.2430	0.2487	0.2534	0.2580	0.2620
Macro-F1	Spectral Clustering	0.1123	0.1320	0.1412	0.1439	0.1526	0.1582	0.1571	0.1588	0.1497
	Social Dimension	0.1001	0.1170	0.1326	0.1508	0.1520	0.1576	0.1639	0.1588	0.1626
	GraRep	0.1508	0.1697	0.1812	0.1891	0.2006	0.2027	0.2073	0.2063	0.2058
	Deepwalk	0.1282	0.1483	0.1622	0.1729	0.1781	0.1772	0.1863	0.1874	0.1843
	LINE	0.1107	0.1331	0.1467	0.1559	0.1672	0.1744	0.1767	0.1771	0.1828
	NetMF	0.1384	0.1662	0.1800	0.1892	0.1950	0.2005	0.2062	0.2101	0.2109
	BoostNE	0.1469	0.1726	0.1867	0.1971	0.2011	0.2035	0.2082	0.2098	0.2098

TABLE IV: Node classification performance comparison on the PPI dataset.

Training Ratio		10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	Spectral Clustering	0.3663	0.3879	0.3977	0.4044	0.4119	0.4109	0.4140	0.4180	0.4191
	Social Dimension	0.2721	0.3094	0.3215	0.3324	0.3326	0.3381	0.3411	0.3377	0.3360
	GraRep	0.3624	0.3781	0.3864	0.3893	0.3925	0.3963	0.3996	0.4002	0.4068
	Deepwalk	0.3358	0.3621	0.3777	0.3865	0.3931	0.3991	0.4042	0.4051	0.4171
	LINE	0.2960	0.3336	0.3557	0.3665	0.3755	0.3828	0.3829	0.3898	0.3938
	NetMF	0.3752	0.3945	0.4051	0.4133	0.4178	0.4202	0.4235	0.4267	0.4305
	BoostNE	0.3908	0.4088	0.4173	0.4242	0.4291	0.4304	0.4322	0.4364	0.4410
Macro-F1	Spectral Clustering	0.2142	0.2344	0.2450	0.2519	0.2587	0.2573	0.2612	0.2608	0.2610
	Social Dimension	0.1503	0.1650	0.1757	0.1805	0.1808	0.1818	0.1841	0.1766	0.1768
	GraRep	0.1968	0.2160	0.2245	0.2280	0.2316	0.2341	0.2382	0.2376	0.2489
	Deepwalk	0.1904	0.2186	0.2347	0.2440	0.2520	0.2581	0.2677	0.2701	0.2684
	LINE	0.1738	0.2003	0.2176	0.2264	0.2363	0.2433	0.2391	0.2508	0.2532
	NetMF	0.2165	0.2358	0.2497	0.2572	0.2615	0.2644	0.2675	0.2686	0.2692
	BoostNE	0.2325	0.2544	0.2674	0.2759	0.2807	0.2844	0.2851	0.2860	0.2852

TABLE V: Node classification performance comparison on the Blogcatalog dataset.

node representations than NetMF. The main reason is that the low-rank assumption of the node connectivity matrix does not always hold in practice and through the multiple stages of low-rank approximation, the node connectivity patterns are better approximated in BoostNE, and further leads more

discriminative node embeddings.

F. Impact of the Number of Embedding Levels k

As shown in the previous subsection, larger embedding level k often results in a smaller approximation error of the

Fig. 3: Impact of the level number k on the embeddings w.r.t. Micro-F1 (similar observations with Macro-F1).

node connectivity matrix. To further investigate how different values of k impact the learned embeddings, we show the node classification performance w.r.t. Micro-F1 and Macro-F1 of BoostNE with different values of k and NetMF in Fig. 3. As can be observed from the figures, when we increase the value of k , the classification performance first increases then reaches its peak and then gradually decreases. The observations are consistent with different portions of training data (30%, 50%, and 70%) and the best classification performance is achieved when k is set as 4 or 8, where the corresponding d_s is 32 or 16. In addition, we also observe that the performance of BoostNE is better than NetMF in a wide range (when k is varied from 2 to 32), which further shows the validity of performing boosted low-rank approximation in learning node embeddings.

V. RELATED WORK

Network Embedding The story of network embedding can be dated back to the early 2000s, when myriad of graph embedding algorithms [8], [27] were developed, as a part of the general dimensionality reduction techniques. Along with this line, we witnessed a surge of factorization based network embedding methods in recent years, with the target to decompose a carefully designed affinity matrix in capturing the 1st-order [8], [18], higher-order [9], [19] node proximity or community structure [2], [7] of the underlying network. Despite their empirical success, the factorization based network embedding methods have at least a quadratic time complexity w.r.t. the number of nodes, prohibiting their practical usage on large-scale networks. The recent advances of network representation learning is largely influenced by the word2vec [11] model in the NLP community. The seminal work of Deepwalk [1] first makes an analogy between truncated random walks on a network and sentences in a corpus, and then learns the embedding representations of nodes with the same principle as word2vec. Typical embedding methods along this line include LINE [12], node2vec [3] and PTE [13]. A recent work found that the embedding methods with negative sampling (e.g., Deepwalk, LINE, PTE, and node2vec) can be unified into a matrix factorization framework with closed-form solutions [14], which bridges the gap between these two families of network embedding methods. Aforementioned methods mainly adopt a shallow model and the expressibility of the learned embedding representations are rather limited. As a remedy, researchers also resort to deep learning techniques [28], [29] to learn more complex and nonlinear mapping functions. Our work is also closely related to recently proposed HARP [30] and MILE [31]. Both of these

two methods attempt to learn node embeddings at different levels by coarsening the original graph into smaller ones, while our proposed BoostNE focuses on the affinity matrix built on the original graph and does not require the graph coarsening step. Additionally, HARP [30] cannot easily adopt other embedding methods (e.g., GraRep and NetMF) as base models, while BoostNE can be generalized to many different network embedding algorithms; MILE [31] tries to improve the scalability of existing network embedding algorithms and refines the embedding from the coarsest graph to the original one with graph convolution network, while BoostNE targets to learn better node embeddings with boosted low-rank matrix approximation. In addition to the raw network structure, real-world networks are often presented with different properties, thus there is a growing interest to learn the embedding representations of networks from different perspectives, such as attributed network [17], [32], [33], heterogeneous networks [28], [34], multi-dimensional networks [35], [36] and dynamic networks [37], [38].

Ensemble-based Matrix Factorization Ensemble methods have shown to be effective in improving the performance of single matrix factorization models, especially in the context of collaborative filtering. DeCoste [39] made one of the first attempts to use ensemble methods to improve the prediction results of a single Maximum Margin Matrix Factorization model. In the sequel, the effectiveness of ensemble models is further validated on the Nystrom method [40] and Divide-and-Conquer matrix factorization [41]. The winner of Netflix Prize [42], [43] advanced the performance of recommendation by capitalizing the advantages of memory-based models and latent factor models. Lee et al. [44] proposed a feature induction algorithm that works in conjunction with stagewise least-squares, and the combination with induced features is superior to existing methods. Most of the existing efforts on matrix factorization are fundamentally based on the prevalent assumption that the given matrix is low-rank, which is often too strong to hold in practice. Lee et al. developed a novel framework LLORMA model [15] to combine the factorization results from multiple locally weighted submatrices, with the assumption that only the local submatrices are low-rank. The success of LLORMA is further extended to other related problems, including social recommendation [23], topic discovery [25], and image processing [45].

VI. CONCLUSIONS AND FUTURE WORK

Network embedding is a fundamental task in graph mining. Recent research efforts have shown that a vast majority of

existing network embedding methods can be unified to the general framework of matrix factorization. Specifically, these methods can be summarized with the following working mechanisms: first, construct the deterministic node connectivity matrix by capturing various types of node interactions, and then apply low-rank approximation techniques to obtain the final node embedding representations. However, the fundamental low-rank assumption of the node connectivity matrix may not necessarily hold in practice, making the resultant low-dimensional node representation inadequate for downstream learning tasks. To address this issue, we relax the global low-rank assumption and propose to learn multiple network representations of different granularity from coarse to fine in a forward stagewise fashion. The superiority of the proposed multi-level network embedding framework over other popular network embedding methods is also in line with the success of gradient boosting framework, where the ensemble of multiple weak embedding representations (learners) leads to a better and more discriminative one (learner).

Future work can be focused on the following two aspects: first we will provide theoretical results to further understand the principles of the boosted NMF approach; second, we plan to investigate methods that can automatically learn the weights of different levels for more meaningful and discriminative embedding representations.

VII. ACKNOWLEDGEMENTS

This material is, in part, supported by the National Science Foundation (NSF) under grant number 1614576.

REFERENCES

- [1] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, pp. 701–710, 2014.
- [2] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, pp. 203–209, 2017.
- [3] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, pp. 855–864, 2016.
- [4] Z. Wang, Z. Jiang, Z. Ren, J. Tang, and D. Yin, "A path-constrained framework for discriminating substitutable and complementary products in e-commerce," in *WSDM*, pp. 619–627, 2018.
- [5] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *arXiv preprint arXiv:1802.00543*, 2018.
- [6] B. Zhang and M. Al Hasan, "Name disambiguation in anonymized graphs using network embedding," in *CIKM*, pp. 1239–1248, 2017.
- [7] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *KDD*, pp. 817–826, 2009.
- [8] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *NIPS*, pp. 585–591, 2002.
- [9] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *CIKM*, pp. 891–900, 2015.
- [10] C. Yang, M. Sun, Z. Liu, and C. Tu, "Fast network embedding enhancement via high order proximity approximation," in *IJCAI*, pp. 19–25, 2017.
- [11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, pp. 3111–3119, 2013.
- [12] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, pp. 1067–1077, 2015.
- [13] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *KDD*, pp. 1165–1174, 2015.
- [14] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *WSDM*, pp. 459–467, 2018.
- [15] J. Lee, S. Kim, G. Lebanon, and Y. Singer, "Local low-rank matrix approximation," in *ICML*, pp. 82–90, 2013.
- [16] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, pp. 1189–1232, 2001.
- [17] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *IJCAI*, pp. 2111–2117, 2015.
- [18] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *WWW*, pp. 37–48, 2013.
- [19] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *KDD*, pp. 1105–1114, 2016.
- [20] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *NIPS*, pp. 2177–2185, 2014.
- [21] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *NIPS*, pp. 556–562, 2001.
- [22] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007.
- [23] H. Zhao, Q. Yao, J. T. Kwok, and D. L. Lee, "Collaborative filtering with social local models," in *ICDM*, pp. 645–654, 2017.
- [24] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning*, vol. 1. Springer, 2001.
- [25] S. Suh, J. Choo, J. Lee, and C. K. Reddy, "L-ensnmf: Boosted local topic discovery via ensemble of nonnegative matrix factorization," in *ICDM*, pp. 479–488, 2016.
- [26] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, no. Aug, pp. 1871–1874, 2008.
- [27] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
- [28] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *KDD*, pp. 119–128, 2015.
- [29] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD*, pp. 1225–1234, 2016.
- [30] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "Harp: Hierarchical representation learning for networks," in *AAAI*, 2018.
- [31] J. Liang, S. Gurukar, and S. Parthasarathy, "Mile: A multi-level framework for scalable graph embedding," *arXiv preprint arXiv:1802.09612*, 2018.
- [32] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *WSDM*, pp. 731–739, 2017.
- [33] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *SDM*, pp. 633–641, 2017.
- [34] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD*, pp. 135–144, 2017.
- [35] Y. Ma, Z. Ren, Z. Jiang, J. Tang, and D. Yin, "Multi-dimensional network embedding with hierarchical structure," in *WSDM*, pp. 387–395, 2018.
- [36] H. Zhang, L. Qiu, L. Yi, and Y. Song, "Scalable multiplex network embedding," in *IJCAI*, 2018.
- [37] J. Li, H. Dani, X. Hu, J. Tang, Y. Chang, and H. Liu, "Attributed network embedding for learning in a dynamic environment," in *CIKM*, pp. 387–396, 2017.
- [38] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *AAAI*, 2018.
- [39] D. DeCoste, "Collaborative prediction using ensembles of maximum margin matrix factorizations," in *ICML*, pp. 249–256, 2006.
- [40] S. Kumar, M. Mohri, and A. Talwalkar, "Ensemble nyström method," in *NIPS*, pp. 1060–1068, 2009.
- [41] L. W. Mackey, M. I. Jordan, and A. Talwalkar, "Divide-and-conquer matrix factorization," in *NIPS*, pp. 1134–1142, 2011.
- [42] R. Bell, Y. Koren, and C. Volinsky, "Modeling relationships at multiple scales to improve accuracy of large recommender systems," in *KDD*, pp. 95–104, 2007.
- [43] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *KDD*, pp. 426–434, 2008.
- [44] J. Lee, M. Sun, S. Kim, and G. Lebanon, "Automatic feature induction for stagewise collaborative filtering," in *NIPS*, pp. 314–322, 2012.
- [45] Q. Yao and J. T. Y. Kwok, "Colorization by patch-based local low-rank matrix completion," in *AAAI*, pp. 1959–1965, 2015.