Unsupervised Multiview Embedding of Node Embeddings

Jia Chen
Dept. of Electrical and Computer Engineering
University of California Riverside
jiac@ucr.edu

Dalia Orozco
Dept. of Electrical and Computer Engineering
University of Texas Rio Grande Valley
dalia.orozco01@outlook.com

Lizeth Figueroa
Dept. of Electrical and Computer Engineering
University of Texas Rio Grande Valley
lizeth.figueroa01@utrgv.edu

Evangelos E. Papalexakis
Dept. of Computer Science and Engineering
University of California Riverside
epapalex@cs.ucr.edu

***Abstract*—In this paper, we propose a comprehensive unsupervised framework that leverages existing and novel multiview learning models, towards obtaining a single node embedding from a collection of node embeddings, combining the best of all worlds. Through extensive experiments, we demonstrate that the proposed multiview node embedding is able to perform on par or better than the best of its constituents and provide reliable performance across downstream tasks including node classification and graph reconstruction.**

***Index Terms*—multiview learning, node embedding, hybrid tensor decomposition, unsupervised learning**

I. Introduction

Graphs are appear in every aspect of life. Transportation networks, e.g., road networks and power lines, capture infrastructure movement or flow among geographical locations. Other examples include scientific paper citation networks, social networks, and brain connectives, to name a few. The interest of performing node embedding on graphs is growing, which learns latent representations of nodes/vertices for a given graph while preserving the neighborhood similarity or connectivity in the original graph. Node embedding is a core step for a lot of downstream tasks.

Long in the purview of researchers, node embedding is a well-established problem and various approaches have been proposed. Most state-of-the-art node embedding techniques assume some measure of *graph smoothness*, thus consider nodes' representations are similar when they are close in the graph. Such type of approaches include DeepWalk [1], Node2Vec [2], HOPE [3], Graph Factorization [4], Walklets [5], Grarep [6], LINE [7], and so on. However, which embedding to choose remains an open problem.

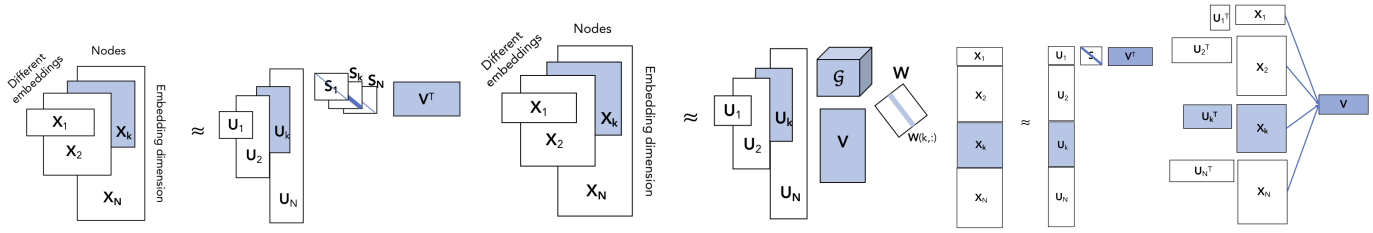
In this paper, we propose to embed node embeddings with a number of multivew learning methids and generate a single embedding by leveraging the complementary advances of different embeddings. Specifically, we use four different multiview learning algorithms to learn a shared node representation among the multiple representations obtained from different embeddings or the same embedding technique with different pre-defined dimensions which are called multiple views. In earlier preliminary work, we use irregular tensor factorization to consolidate multiple embeddings obtained from a single node embedding (DeepWalk) with various dimensions for the node representations [8]. Different from [8], our proposed framework considers various types of node embeddings as different views providing a more comprehensive ensemble node embedding framework. Our main contributions are:

- We propose four ensemble node embedding schemes;
- We develop a new tensor model HyTuck2 that is well-suited for the task, in the sense that it is able to capture non low-rank trilinear structure;
- We extensively evaluate our framework on three datasets and test the effectiveness of our approach in classification and graph reconstruction tasks.

II. Proposed Method

Given a graph $\mathbf{G} := \{\mathcal{V}, \mathcal{E}\}$ with $\mathcal{V} := \{a_1, a_2, ..., a_m\}$ collecting all the m nodes and $\mathcal{E}$ indicating the edge set where the (binary or weighted) adjacency matrix is $\mathbf{A} \in \mathbb{R}^{m \times m}$, we consider the problem of learning the node representations $\mathbf{V} \in \mathbb{R}^{m \times d}$ while preserving the node similarities in $\mathbf{G}$ from higher-dimensional space $\mathbb{R}^m$ to a lower-dimensional space $\mathbb{R}^d$ with $d \ll m$. Toward this end, we propose a two-step multiview node embedding scheme by first running the existing node embedding algorithms, e.g., DeepWalk, Node2Vec, and LINE, to get multiview data, denoted as $\{\mathbf{X}_n \in \mathbb{R}^{d_n \times m}\}_{n=1}^N$ where d_n is the pre-defined dimension of the n-th embedding/view and N is the total number of embeddings, and then applying multiview learning models to fuse the N views into a shared node embedding $\mathbf{V}$. Note that, datasets from the same node embedding with different dimensions are seen as different views. Below we describe the multiview learning models of our framework.

Research was supported by NSF under CAREER grant no. IIS 2046086 and CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS) grant no. 2112650, and UCR Regents Faculty Fellowship.



(a) PARAFAC2 (b) Proposed HYTUCK2 (c) Stacked SVD (d) MCCA

Fig. 1: Multiview learning models. Each sub-figure also shows the way in which each view $\mathbf{X}_n$ is related to the model in shaded color.

A. Projection model

Canonical correlation analysis (CCA) finds the shared representation of two datasets by projecting them into the same space and forcing the two projected datasets to be close in the Euclidean space or the correlation to be maximized [9]. Multiview (M) CCA generalizes CCA from dealing with two datasets jointly to arbitrary (≥ 2) number of datasets [10].

Given the sought multiview embeddings $\{\mathbf{X}_n\}_{n=1}^N$, MCCA searches for the shared node embedding $\mathbf{V}$ by solving

$$\min_{\{\mathbf{U}_n\}_{n=1}^N, \mathbf{V}} \quad \sum_{n=1}^{N} \left\| \mathbf{U}_n^\top \bar{\mathbf{X}}_n - \mathbf{V}^\top \right\|_F^2 \tag{1}$$

under the constraint that $\mathbf{V}^\top \mathbf{V} = \mathbf{I}$, where $\bar{\mathbf{X}}_n$ is the centered data of $\mathbf{X}$, $\{\mathbf{U}_n \in \mathbb{R}^{d_n \times d}\}_{n=1}^N$ are projection matrices, and the constraint is imposed to avoid a trivial solution; see Fig. 1-(d) for further demonstration. The optimal $\mathbf{V}$ is obtained by computing the eigenvalue decomposition on the matrix $\sum_{n=1}^N \bar{\mathbf{X}}_n^\top (\bar{\mathbf{X}}_n \bar{\mathbf{X}}_n^\top)^{-1} \bar{\mathbf{X}}_n$, and the columns of $\mathbf{V}$ are the d eigenvectors corresponding to the top-n eigenvalues. The optimal projection matrices are given by $\{\mathbf{U}_n = (\bar{\mathbf{X}}_n \bar{\mathbf{X}}_n^\top)^{-1} \bar{\mathbf{X}}_n \mathbf{V}\}_{n=1}^N$.

B. Factorization models

From factorization perspective, we consider three multiview learning models: stacked SVD and two irregular tensor decomposition methods. Intuitively, we can concatenate all the individual embeddings generating a higher-dimensional data matrix $\mathbf{X} := [\mathbf{X}_1^\top, \mathbf{X}_2^\top, ..., \mathbf{X}_N^\top]^\top \in \mathbb{R}^{(d_1+d_2+...+d_N) \times m}$ and then perform SVD on $\mathbf{X}$, i.e., $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, to get a lower-dimensional representation $\mathbf{V}$, which is referred to as stacked SVD; see Fig. 1-(c).

Given embeddings $\{\mathbf{X}_n\}_{n=1}^N$, irregular tensor decomposition method, namely PARAFAC2 (shown in Fig. 1-(a)) is used to extract the ensemble embedding $\mathbf{V}$, which is boiled down to tackling the following constrained minimization problem

$$\begin{aligned} \min_{\{\mathbf{U}_n \in \mathbb{R}^{d_n \times d}, \mathbf{S}_n \in \mathbb{R}^{d \times d}\}_{n=1}^N, \mathbf{V}} \quad & \sum_{n=1}^{N} \|\mathbf{X}_n - \mathbf{U}_n \mathbf{S}_n \mathbf{V}^\top\|_\mathbf{F}^\mathbf{2} \\ \text{s. to} \quad & \mathbf{U}_n = \mathbf{Q}_n \mathbf{H},\ \mathbf{Q}_n^\top \mathbf{Q}_n = \mathbf{I},\ \forall n \end{aligned} \tag{2}$$

where $\mathbf{S}_n$ is a diagonal matrix whose diagonal entries indicate the importance of latent components (a.k.a., the columns of $\mathbf{V}$), factor matrix $\mathbf{U}_n$ is decomposed into two matrices, $\mathbf{Q}_n$ that has orthonormal columns and $\mathbf{H}$ which is invariant regardless of n. Problem (2) can be solved by alternating least squares (ALS) approach [11].

Furthermore, we develop a new tensor model HYTUCK2, shown in Fig. 1-(b), to fulfill such multiview learning task, which can be seen as the generalization of the Tucker model [12] (which is applicable to regular tensors), to the paradigm of PARAFAC2 which applies to irregular tensors where there is a mismatch in one of the modes. We define the slice representation of HYTUCK2 as follows:

$$\mathbf{X}_n \approx \mathcal{G} \times_1 \mathbf{U}_n \times_2 \mathbf{V} \times_3 \mathbf{W}(n,:)$$

with similar constraints as PARAFAC2: $\mathbf{U}_n = \mathbf{Q}_n \mathbf{H}$, $\mathbf{Q}_n^\top \mathbf{Q}_n = \mathbf{I}$, $\forall n$, and where $\mathcal{G}$ is a core tensor, which captures multi-way interactions between the different modes. An advantage of HYTUCK2 against PARAFAC2 is its ability to naturally express structure that does not adhere to the low-rank trilinear model that PARAFAC2 assumes.

In order to fit the HYTUCK2 model, we follow an Alternating Least Squares (ALS) approach, as the one that is traditionally used for PARAFAC2 [11]. The main difference in the case of HYTUCK2 is that the projected tensor $\mathcal{Y}$ from Alg. 2 in [11], whose slices are constructed as $\mathbf{Y}_n = \mathbf{Q}_n^\top \mathbf{X}_n$, is now decomposed according to the Tucker-3 model, instead of the CP model. For computational efficiency, we can fit a single-shot approximation of the Tucker-3 model via the Higher-Order or Multilinear SVD (HOSVD or MLSVD), instead of the more costly ALS-based Higher-Order Orthogonal Iteration (HOOI) [12]. In experiments, we observed that both solutions yielded numerically almost identical results, thus, we opt for the more economical approach.

III. EXPERIMENTAL EVALUATION

We evaluate our approach on two tasks: node classification, and graph reconstruction. Three real-world datasets are considered: (1) **Blogcatalog** is social relationship network data provided by blogger authors with labels representing the topic categories, which consists of $10,312$ nodes, $333,983$ edges, and 39 labels [13]; (2) **Cora** dataset consisting of $2,708$ nodes, $5,429$ edges, and 7 labels is a citation network for scientific publications [14]; and (3) **Wiki** dataset is a co-occurrence network of words showing up in the first million bytes of

the Wikipedia dump with $2,405$ nodes, $17,981$ edges, and 19 labels [15]. The number of ALS iterations in both HYTUCK2 and PARAFAC2 are set to be 15 in Secs. III-A and III-B. In all experimental results, the best performance appears in bold. We use the code provided by [11] for PARAFAC2, and make our code for fitting HYTUCK2 publicly available[1].

A. Node classification

We evaluate the embeddings on two node classification scenarios: **Scenario 1**: some views provide good classification results; **Scenario 2**: none of the views provide good performance. In each scenario, we compare our proposed multiview node embeddings techniques with individual embedding models including DeepWalk [1], Node2Vec [2], HOPE [3], LINE [7], graph representation (Graphrep) [6], and/or graph factorization (Graphfac) [4]. We use OpenNE[2] to obtain all the single views from the existing node embedding techniques. We randomly choose 90% of the data as traning data and the rest are used as test. For both individual embedding models and ensemble embeddings models, a one-vs-rest logistic regression implemented by LibLinear [16] is used to carried out the classification and we use the code provided online[3] . We repeat this process 50 times, and report the average Micro-F1 and Macro-F1 scores with their corresponding standard deviations.

To test the effectiveness of the proposed schemes in **Scenario 1** using Cora dataset, in the middle part of Table I, we show the classification performance of $m = 15$ single views including implementing various state-of-the-art node embeddings techniques with different embedding dimensions which are specified after the model names, e.g., Node2Vec-128. The Micro-F1 and Macro-F1 scores from the proposed four multiview embedding models are shown on the bottom of Table I. Further, we investigate the classification results of different models in **Scenario 2**, a more challenging case when $m = 7$ views are used; see Table II. After combining the results in Tables I and II, one can conclude that when single view(s) can provide promising performance, our multiview models perform competitively; while when all the views are performing very poorly using multiview models the performance has been improved significantly. Similar conclusions are drawn when testing Blogcatalog and Wiki datasets, see results for **Scenario 1** in Tables III and IV and results for **Scenario 2** in Tables V and VI.

B. Graph reconstruction

We test the quality of our multiview node embeddings in graph reconstruction. Given the embedding $\mathbf{V}$, we use k-nearest neighbors to recover the graph $\mathbf{G}$ via taking the following three steps:

- **Step 1:** Calculate the Euclidean distance between two nodes and assign the inverse of the distance as the similarity between them, i.e., the (i, j)-th entry of the similarity matrix is $\mathbf{A}_0(i, j) := \frac{1}{\|\mathbf{v}_i - \mathbf{v}_j\|_2 + \epsilon}$ where $\mathbf{v}_i$ is the i-th column of $\mathbf{V}$ and $\epsilon = 2.2 \times 10^{-16}$ is used to avoid $\mathbf{A}_0(i, j) = \infty$.
- **Step 2:** Sparsify $\mathbf{A}_0$ to $\mathbf{A}_1$ by preserving the k-nearest neighbors of each node and setting the rest entries as zeros.
- **Step 3:** Sort weights in $\mathbf{A}_1$ in a descending order and set the top-T weights to be 1 with the rest to be 0 using which as the adjacency matrix forming the reconstructed graph $\hat{\mathbf{G}}$, where T is the number of edges in the orginal graph $\mathbf{G}$.

To evaluate the quality of the reconstructed graph, we adopt the DeltaCon method [17] to quantify the similarity in connectivity between two graphs: $\mathbf{G}$ and $\hat{\mathbf{G}}$. Such graph similarity score is in the range of $[0, 1]$, and 0 means totally different graphs, while 1 means identical graphs. We set $k = 500$ when testing Blogcatalog data and $k = 100$ when testing Cora and Blogcatalo data to make sure that after Steps 1 and 2 there are enough edges with nonzero weights for Step 3. Note that when reconstructing the graph using a single view data from the existing node embedding technique, $\mathbf{V}$ is substituted with the corresponding embedding $\mathbf{X}_n$. We consider two cases:**Case 1:** some views have good graph reconstruction performance

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
Node2Vec-32	.80 ± .028	.79 ± .034	**DeepWalk-16**	.73 ± .027	.69 ± .033
Node2Vec-128	.81 ± .027	.80 ± .031	**DeepWalk-32**	.78 ± .024	.77 ± .027
Node2Vec-200	.82 ± .023	.82 ± .027	**DeepWalk-200**	**.85** ± .018	**.84** ± 0.02
LINE-32	.36 ± .026	.16 ± .032	**HOPE-16**	.45 ± .031	.28 ± .033
LINE-200	.50 ± .026	.43 ± .032	**HOPE-64**	.57 ± .026	.52 ± .016
			HOPE-200	.68 ± .026	.66 ± .032
Grafac-32	.50 ± .02	.43 ± .023	**Grarep-32**	.72 ± .029	.67 ± .014
Grafac-200	.58 ± .022	.56 ± .025	**Grarep-200**	.77 ± .02	.75 ± .022
StackSVD-200	**.85** ± .021	**.84** ± .025	**PARAFAC2-200**	**.85** ± .015	**.84** ± .018
MCCA-200	.83 ± .022	.82 ± .023	**HYTUCK2-200**	**.85** ± .02	**.84** ± .023

TABLE I: F1 scores in **Scenario 1** using **Cora** data. In HYTUCK2-200, $p = 200$, $d = 200$, and $q = 15$. In HYTUCK2-300, $p = 400$, $d = 300$, $q = 5$.

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
HOPE-16	.45 ± .031	.28 ± .033	**HOPE-200**	.68 ± .026	.66 ± .032
LINE-32	.36 ± 0.026	.16 ± .032	**LINE-64**	.39 ± .027	.22 ± .027
LINE-200	.50 ± .026	.43 ± .032	**Grarep-16**	.64 ± .029	.54 ± .025
Grafac-64	.53 ± .029	.50 ± .031			
StackSVD-200	.77 ± .024	.75 ± .028	**PARAFAC2-200**	.77 ± .024	**.76** ± .028
MCCA-200	.73 ± .026	.71 ± .027	**HYTUCK2-200-1**	**.78** ± .026	**.76** ± .03
			HYTUCK2-200-2	**.78** ± .022	**.76** ± .025

TABLE II: F1 in **Scenario 2** using **Cora** data. $p, d = 200$, $q = 7$ for HYTUCK2-200-1. HYTUCK2-200-2, $p = 500$, $d = 200$, $q = 10$.

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
Node2Vec-32	.40 ± .010	.23 ± .009	**DeepWalk-32**	.40 ± .013	.23 ± .009
Node2Vec-64	.41 ± .013	.26 ± .014	**DeepWalk-64**	**.42** ± .013	.27 ± .014
Node2Vec-128	.41 ± .013	.27 ± .014	**DeepWalk-200**	**.42** ± .012	**.29** ± .014
Node2Vec-200	.40 ± .013	.27 ± .014			
LINE-32	.27 ± .008	.11 ± .008	**HOPE-32**	.22 ± .01	.07 ± .007
LINE-200	.38 ± .01	.23 ± .009	**HOPE-200**	.31 ± .009	.15 ± .01
Grafac-32	.21 ± .013	.05 ± .004	**Grarep-32**	.23 ± .013	.08 ± .015
Grafac-200	.30 ± .011	.12 ± .008	**Grarep-200**	.37 ± .009	.21 ± .008
StackSVD-200	.41 ± .009	.27 ± .011	**PARAFAC2-200**	.41 ± .01	.27 ± .013
MCCA-200	**.42** ± .013	**.29** ± .011	**HYTUCK2-200**	.41 ± .014	.28 ± .011

TABLE III: Micro-F1 and Macro-F1 in **Scenario 1** using **Blogcatalog** data. In HYTUCK2, the dimensions are set as $p = 200$, $d = 200$, and $q = 15$.

[1]https://www.cs.ucr.edu/~epapalex/src/HyTuck2-public-code.zip
[2]https://github.com/thunlp/OpenNE
[3]https://github.com/phanein/deepwalk

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
Node2Vec-32	.62 ± .032	.45 ± .05	**DeepWalk-32**	.64 ± .024	.48 ± .044
Node2Vec-64	.65 ± 0.029	.49 ± .034	**DeepWalk-128**	.69 ± .032	.57 ± .036
Node2Vec-200	.66 ± .032	.51 ± .052	**DeepWalk-200**	**.71** ± .029	**.59** ± 0.021
LINE-32	.36 ± .03	.24 ± .025	**HOPE-32**	.49 ± .03	.33 ± .018
LINE-200	.58 ± .033	.41 ± .027	**HOPE-128**	.60 ± .024	.43 ± .034
			HOPE-200	.61 ± .027	.45 ± .026
Grafac-32	.51 ± .032	.34 ± .024	**Grarep-32**	.54 ± .03	.34 ± .018
Grafac-200	.57 ± .036	.41 ± .033	**Grarep-200**	.64 ± .032	.47 ± .039
StackSVD-200	.70 ± .025	.58 ± .044	**PARAFAC2-200**	.70 ± .03	**.58** ± .042
MCCA-200	.69 ± .024	.57 ± .044	**HyTuck2-200-1**	.70 ± .032	**.59** ± .06
MCCA-250	.70 ± .03	.59 ± .028	**HyTuck2-200-2**	**.71** ± .028	**.59** ± .034

TABLE IV: Micro-F1 and Macro-F1 in **Scenario 1** using **Wiki** data. In HyTuck2-200-1, the dimensions are set as $p = 200$, $d = 200$, and $q = 15$. In HyTuck2-200-2, the dimensions are set as $p = 200$, $d = 200$, and $q = 10$.

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
HOPE-32	.22 ± .01	.07 ± .007	**HOPE-200**	.31 ± .009	.15 ± .01
Grafac-32	.21 ± .013	.05 ± .004	**Grafac-200**	.30 ± .011	.12 ± .008
LINE-32	.27 ± .008	.11 ± .008	**Grarep-32**	.23 ± .013	.08 ± .015
StackSVD-200	.34 ± .013	.18 ± .011	**PARAFAC2-200**	**.35** ± .011	**.20** ± .01
MCCA-200	.34 ± .014	**.20** ± .011	**HyTuck2-200**	.34 ± .011	**.20** ± .009

TABLE V: F1 score in **Scenario 2** using **Blogcatalog** data. In HyTuck2, $p = 200$, $d = 200$, and $q = 6$.

Models	Micro-F1	Macro-F1	Models	Micro-F1	Macro-F1
HOPE-32	.49 ± .03	.33 ± .018	**LINE-32**	.36 ± .03	.24 ± .025
DeepWalk-16	.58 ± .026	.40 ± .026	**LINE-128**	.54 ± .032	.36 ± .018
Grarep-32	.54 ± .03	.34 ± .018	**LINE-200**	.58 ± .033	.41 ± .027
Grafac-32	.51 ± .032	.34 ± .024			
StackSVD-200	**.66** ± .028	**.53** ± .041	**PARAFAC2-200**	.65 ± .024	.52 ± .039
MCCA-200	.62 ± .026	.51 ± .049	**HyTuck2-200**	.64 ± .029	.52 ± .042

TABLE VI: F1 scores in **Scenario 2** using **Wiki** data. In HyTuck2-200, the we set $p = 200$, $d = 200$, and $q = 7$.

in terms of high graph similarity scores; **Case 2:** none of the views have good performance. We present the graph similarity scores using the Cora dataset in the above two cases for different ratios of training data ranging from 10% to 90%. The results show that MCCA and HyTuck2 perform similarly to the best views in **Case 1**; see Table VII, and remarkably better than the best views in **Case 2**; see Table VIII. Similar conclusion can be drawn when testing Blogcatalog dataset, see the results in IXX.

C. Discussions

1) HyTuck2 *versus PARAFAC2:* We study the ability of each model to capture meaningful structure in the data, especially as a function of the number of ALS iterations necessary in order to do so. Specifically, we compare the fit (as a percentage of the variation explained by each model defined in (3)) of HyTuck2 and PARAFAC2 (whose fit is defined in (3) after substituting the numerator with the objective in (2)) as a function of the number of ALS iterations, on the Wiki dataset. Our results are shown in Fig. 2, where we observe that HyTuck2 with 5 ALS iterations is able to achieve the same fit as PARAFAC2 with 30 iterations, whereas 1 iteration of HyTuck2 is also extremely close to the 30 iterations of PARAFAC2. This outlines the ability of HyTuck2 to better capture the structure in the data, more efficiently.

Models	10%	20%	30%	40%	50%	60%	70%	80%	90%
Node2Vec-32	49	52	53	55	59	59	60	62	66
Node2Vec-128	37	39	40	41	43	43	45	46	49
Node2Vec-200	37	38	40	41	43	43	44	46	48
DeepWalk-16	52	55	56	58	62	63	64	66	70
DeepWalk-32	54	57	59	61	65	66	67	69	73
DeepWalk-200	**56**	**59**	**61**	**64**	**68**	**68**	**69**	**71**	**75**
LINE-32	46	49	51	53	57	57	59	61	66
LINE-200	54	58	60	62	66	66	67	69	74
HOPE-16	22	23	24	25	26	26	26	27	28
HOPE-64	23	24	24	25	26	26	27	27	28
HOPE-200	23	24	24	25	26	26	27	27	29
Grafac-32	50	52	54	56	59	60	61	63	67
Grafac-200	53	56	58	60	63	63	65	67	71
Grarep-32	23	23	24	25	26	26	27	27	28
Grarep-200	23	23	24	25	26	26	26	27	29
StackSVD-200	53	57	59	61	65	65	66	69	73
PARAFAC2-200	52	55	57	59	63	63	64	66	71
MCCA-200	54	57	59	61	65	66	67	69	73
MCCA-300	55	58	**61**	63	67	67	**69**	70	**75**
HyTuck2-200	54	57	59	62	66	66	67	69	74
HyTuck2-300	55	**59**	**61**	63	**68**	**68**	**69**	**71**	**75**

TABLE VII: Graph similarity scores (%) between the original and reconstructed graphs for different ratios of training data in **Case 1** using **Cora** data. In HyTuck2-200,$p = 200$, $d = 200$, and $q = 15$. In HyTuck2-300, $p = 400$, $d = 300$, and $q = 5$.

Models	10%	20%	30%	40%	50%	60%	70%	80%	90%
LINE-32	46	49	51	53	57	57	59	61	66
LINE-64	48	51	53	55	59	60	61	63	68
LINE-200	54	58	60	62	66	66	67	69	73
HOPE-16	22	23	24	25	26	26	26	27	28
HOPE-200	23	24	24	25	26	26	27	28	29
Grarep-16	23	23	24	25	26	26	26	27	28
Grafac-64	52	55	57	59	62	63	64	66	70
StackSVD-200	48	50	52	54	58	58	60	62	67
PARAFAC2-200	49	51	53	55	58	59	59	62	66
MCCA-200	**58**	**62**	**64**	**67**	**71**	**71**	**72**	**73**	**77**
HyTuck2-200-1	55	58	61	63	67	68	69	70	74
HyTuck2-200-2	55	59	61	63	68	68	69	71	75

TABLE VIII: Graph similarity scores () between the original and reconstructed graphs for different ratios of training data in **Case 2** using **Cora** data. In HyTuck2-200-1, $p = 200$, $d = 200$, and $q = 7$. In HyTuck2-200-2, $p = 500$, $d = 200$, and $q = 10$.

Models	10%	20%	30%	40%	50%	60%	70%	80%	90%
Node2Vec-32	**.29**	**.31**	**32**	**33**	**35**	**35**	**36**	**38**	**41**
Node2Vec-64	26	27	28	29	31	31	32	33	36
Node2Vec-128	22	23	24	25	26	26	27	28	30
Node2Vec-200	20	20	21	22	23	23	24	25	27
DeepWalk-32	29	30	31	32	34	35	36	37	40
DeepWalk-64	26	28	29	29	31	31	32	33	36
DeepWalk-200	24	24	25	26	27	27	28	29	30
LINE-32	21	22	23	24	26	27	28	29	32
LINE-200	18	19	20	21	22	23	23	25	28
HOPE-32	13	14	14	15	16	16	17	18	19
HOPE-200	11	12	12	13	14	14	14	15	16
Grafac-32	27	28	29	30	32	32	33	35	37
Grafac-200	23	23	24	25	26	26	27	28	30
Grarep-32	23	24	25	26	28	29	30	31	34
Grarep-200	25	27	28	29	31	32	33	35	38
StackSVD-200	17	18	19	20	22	22	23	24	27
PARAFAC2-200	24	25	26	27	29	30	31	32	35
MCCA-200	28	30	30	32	34	34	35	36	39
HyTuck2-200	27	28	29	30	32	32	33	35	37

TABLE IX: Graph similarity scores (%) between the original and reconstructed graphs for different ratios of training data in **Case 1** using **Blogcatalog** data. In HyTuck2-200, the dimensions are set as $p = 200$, $d = 200$, and $q = 15$.

$$\text{Fit} = 100 - \frac{\sum_{n=1}^{N} ||\mathbf{X}_n - \mathcal{G} \times_1 \mathbf{U}_n \times_2 \mathbf{V} \times_3 \mathbf{W}(n,:)||_F^2}{\sum_{n=1}^{N} ||\mathbf{X}_n||_F^2} \times 100 \quad (3)$$

Models	10%	20%	30%	40%	50%	60%	70%	80%	90%
HOPE-32	13	14	14	15	16	16	17	18	19
HOPE-200	11	12	12	13	14	14	14	15	16
LINE-32	21	23	23	24	26	27	28	29	33
Grarep-32	23	24	25	26	28	29	30	31	34
Grafac-32	27	28	29	30	32	32	33	35	37
Grafac-200	23	23	24	25	26	26	27	28	30
StackSVD-200	16	16	17	18	19	19	20	22	24
PARAFAC2-200	23	24	25	26	28	29	30	31	35
MCCA-200	**30**	**31**	**32**	**33**	**35**	**35**	**36**	**38**	**40**
HYTUCK2-200	27	29	30	31	33	33	34	36	38

TABLE X: Graph similarity scores (%) between the original and reconstructed graphs for different ratios of training data in **Case 2** using **Blogcatalog** data. In HYTUCK2-200, the dimensions are set as $p = 200$, $d = 200$, and $q = 6$.

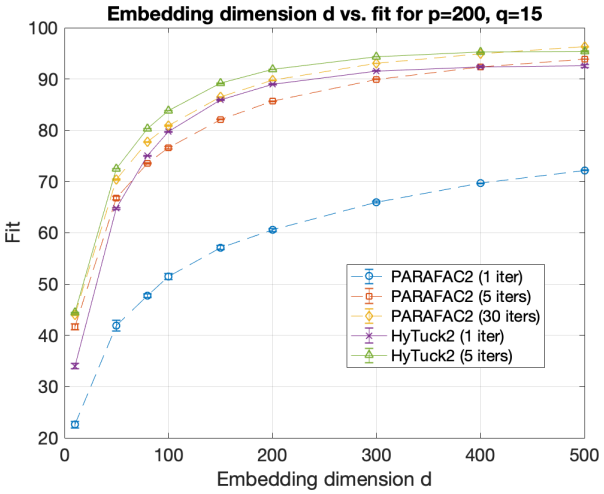


Fig. 2: HYTUCK2 vs. PARAFAC2

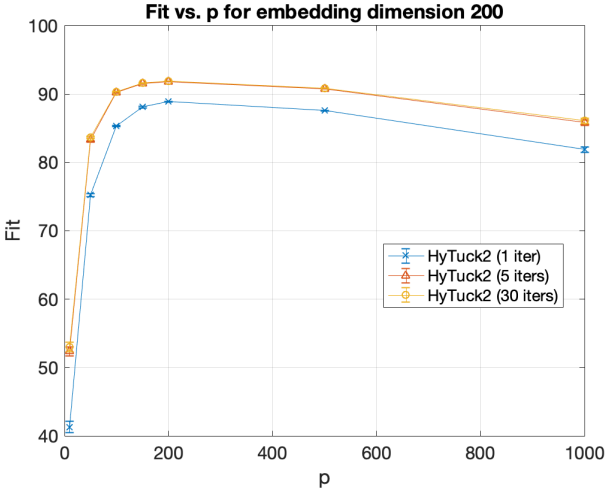


Fig. 3: HYTUCK2 sensitivity analysis

2) Fine-tuning HYTUCK2*:* We investigate how the first dimension p influences the fit of HYTUCK2 when fixing the remaining two dimensions. In Fig. 3 we use the 15 views of the Wiki dataset and we plot the fit versus p when fixing $d = 200$ and $d = 500$ accordingly, which shows that the fit increases with p growing and then decreases due to overfitting.

IV. CONCLUSIONS

In this paper we introduce a comprehensive multivew framework for embedding different node embeddings. To that end, we introduce a novel irregular tensor decomposition model HYTUCK2, which is able to capture structure that existing models cannot. Through extensive experimentation we demonstrate that our proposed multiview framework produces embeddings that are on-par with the best-performing single view, if there is a view that achieves state-of-the-art performance, or performing better than the best-performing view, if no views achieve state-of-the-art.

REFERENCES

[1] 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*, 2014, pp. 701–710.

[2] 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*, 2016, pp. 855–864.

[3] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016.

[4] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 37–48.

[5] B. Perozzi, V. Kulkarni, and S. Skiena, "Walklets: Multiscale graph embeddings for interpretable network classification," *arXiv preprint arXiv:1605.02115*, 2016.

[6] 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*, 2015, pp. 891–900.

[7] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding." in *WWW*. ACM, 2015.

[8] J. Chen and E. E. Papalexakis, "Ensemble node embeddings using tensor decomposition: A case-study on deepwalk," in *1st Workshop on Multi-Source Data Mining, ICDM*, 2020.

[9] H. Hotelling, "Relations between two sets of variates," in *Breakthroughs in statistics*. Springer, 1992, pp. 162–190.

[10] P. Horst, *Generalized canonical correlations and their application to experimental data*. Journal of clinical psychology, 1961, no. 14.

[11] I. Perros, E. E. Papalexakis, F. Wang, R. Vuduc, E. Searles, M. Thompson, and J. Sun, "Spartan: Scalable parafac2 for large & sparse data," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 375–384.

[12] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, 2009.

[13] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 817–826.

[14] L. Getoor, "Link-based classification," in *Advanced methods for knowledge discovery from complex data*. Springer, 2005, pp. 189–207.

[15] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003, pp. 252–259.

[16] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *the Journal of machine Learning research*, vol. 9, pp. 1871–1874, 2008.

[17] D. Koutra, J. T. Vogelstein, and C. Faloutsos, "Deltacon: A principled massive-graph similarity function," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 162–170.