Heterogeneous Hyper-Network Embedding

Inci M. Baytas[1], Cao Xiao[2], Fei Wang[3], Anil K. Jain[1], and Jiayu Zhou[1]
[1]*Computer Science and Engineering, Michigan State University, East Lansing, MI, USA*
[2]*AI for Healthcare, IBM Research, Cambridge, MA, USA*
[3]*Healthcare Policy and Research, Weill Cornell Medical School, Cornell University, New York, NY, USA*
Email: baytasin@msu.edu, cxiao@us.ibm.com, few2001@med.cornell.edu, jain@cse.msu.edu, jiayuz@msu.edu

***Abstract*—Heterogeneous hyper-networks is used to represent multi-modal and composite interactions between data points. In such networks, several different types of nodes form a hyperedge. Heterogeneous hyper-network embedding learns a distributed node representation under such complex interactions while preserving the network structure. However, this is a challenging task due to the multiple modalities and composite interactions. In this study, a deep approach is proposed to embed heterogeneous attributed hyper-networks with complicated and non-linear node relationships. In particular, a fully-connected and graph convolutional layers are designed to project different types of nodes into a common low-dimensional space, a tuple-wise similarity function is proposed to preserve the network structure, and a ranking based loss function is used to improve the similarity scores of hyperedges in the embedding space. The proposed approach is evaluated on synthetic and real world datasets and a better performance is obtained compared with baselines.**

***Keywords*-Heterogeneous hypergraph, network embedding, similarity learning**

I. Introduction

Many real-world problems can be represented by networks, such as social networks, biological and brain networks. Analyzing and modeling the *node* relationships in a network facilitate machine learning tasks on graph structured data, such as node classification and link prediction. In particular, network embedding aims to learn distributed node representations by preserving the underlying network structure. *Edge* structure is also important to decipher the interactions between the nodes. In a standard network, edges represent pairwise relationships between two nodes, e.g., pairwise connections in a social network. On the other hand, some problems consider interactions between several nodes, such as author-paper-conference and the disease-medication-side effect networks [1], [2]. In particular, biomedical networks, such as a patient-drug-disease network, comprises of complex interactions among entities of different modalities [3]. Heterogeneous hyper-network embedding learns node representations for such complex networks by capturing the composite biomedical interactions.

In heterogeneous information networks, composite interactions are defined as tuple-wise relationships between several different types of nodes. The composite interactions can be decomposed into pairwise connections between fixed number of cross-type nodes, e.g., meta-paths. A hyperedge can also be defined as an event and sub-events can be sampled with a fixed number of nodes [2]. However, such approaches ignore the tuple-wise interactions and thus limit the flexibility of the embedding learning. In this paper, a deep heterogeneous hyper-network embedding approach is proposed, where the number of nodes and their types are not fixed in a hyperedge. The proposed approach, coined *HHNE*, learns low-dimensional embeddings for each node while preserving the composite relations defined by the hyperedges in an attributed hyper-network. Contributes of the proposed HHNE model are as follows:

Cross-type and non-linear embedding: Attributes of different types of nodes do not necessarily share a common representation, yet they are semantically related. Therefore, we designed an embedding layer including non-linear fully-connected layer along with a spatial graph convolutional network (GCN) to learn a common latent space, where a similarity score can be computed between cross-type nodes.

Tuple-wise similarity: A tuple-wise similarity function, which does not decompose the hyperedge into predefined meta-paths, is simultaneously learned with the embedding layer. HHNE optimizes the proposed model based on the hypothesis that the tuple-wise similarity of a hyperedge embedding should be higher than that of unrelated nodes. Thus, HHNE ensures that the composite interactions are preserved in the embedding space without setting a limit to the number of nodes in the hyperedge.

Loss function: The proposed model is optimized by penalizing lower rankings of the tuple-wise similarity scores of hyperedges compared with the scores of unrelated nodes. In practice, strictly assuming known interactions as positive labels and the unknown ones as negative labels would ignore the undiscovered relationships. Therefore, the loss function of HHNE does not apply a binary classification scheme, but a ranking approach. Experiments are conducted on synthetic and real-world datasets for quantitative tasks and HHNE is observed to perform better compared with baseline state-of-the-art network embedding methods.

II. Related Work

Embedding learning has always drawn attention in the literature [4]. Some of the embedding learning approaches in-

978-1-5386-9159-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDM.2018.00104

clude truncated random walks with online learning [5], modularized non-negative matrix factorization (M-NMF) [6], and semi-supervised learning for networks with outliers [7]. Deep learning is also used for network embedding, such as a deep embedding learning approach considering both content and relational information for heterogeneous networks [1], a deep random surfing model to capture the structural information [8], a CNN to learn feature representation for arbitrary graphs [9], and a structural semi-supervised deep embedding model to preserve first and second order proximities [10]. Attributed networks are also addressed [11] where both structural and attribute proximities need to be preserved. Some deep models divide attributed graphs into graphlets to learn node and edge features [12]. Distribution of the attributes and topological structure are also important for network embedding [13].

Network embedding approaches mentioned so far focus on pairwise links, which are not sufficient to model complex real world problems. On the other hand, heterogeneous information networks, where various type of nodes are interacting each other, provide an important opportunity to model complex real-world problems. For instance, semi-supervised representation learning [14] and meta-path based embedding learning [15], [16] are commonly utilized in heterogeneous text networks. In addition, heterogeneous hypergraph embedding can also be used to address document recommendation [3] and to preserve contextual information [17]. Recently, heterogeneous hyper-network embedding approach including deep auto-encoders was proposed [18]. However, these studies rely on predefined number and type of nodes in one hyperedge, and the similarity computations comprise of pairwise similarities in meta-paths rather than tuple-wise similarity of a hyperedge.

III. Method

In this section, background information for heterogeneous hyper-network embedding is given and the proposed approach is presented.

A. Background

1) Heterogeneous hyper-network: A hyper-network can be defined as a network where the links between its nodes are defined in a tuple-wise manner. A formal definition of heterogeneous hyper-network is given below.

Definition III.1. A heterogeneous hyper-network is defined as a hypergraph $\mathbf{G} = (E, V)$ with a node set $V = V_1 \bigcup \cdots \bigcup V_K$, which is a union of node sets of K types, and a hyperedge set $E = \{e_1, \cdots, e_{|E|}\}$, where each hyperedge e_i contains more than two different types of nodes.

Hyperedges can be homogeneous and heterogeneous when they contain the same type and different types of nodes, respectively. In real world problems, a hyperedge usually represents semantically related group of nodes. Furthermore, hyperedges model entities with multiple modalities. For instance, in a drug network, each hyperedge contains various modalities of a drug such as its chemical structure, indications and associated ADRs. Similarly, an ADR reporting network contains hyperedges, where each report is represented by one patient, several drugs and their ADRs that the patient experiences. Thus, constructing a hyper-network for datasets with various modalities assists an analysis of the complex relationships of different types of information. On the other hand, learning a distributed representation that can capture the underlying structure of such networks, known as *network embedding*, is crucial to carry out the network analysis tasks.

2) Network embedding: Network embedding aims to learn a low dimensional representation for each node while preserving the local and global network structures in the embedding space such that similarity of the node embeddings in an edge should be higher than that of unrelated nodes. In heterogeneous hyper-networks, different types of nodes may lie in different spaces. For this reason, network embedding should project each node into a common space. Furthermore, tuple-wise similarities should be defined to be able to preserve the composite node relationships. In real world applications, the composite interactions have a non-linear nature that the embedding learning needs to address. Therefore, in this study, a deep embedding approach along with a tuple-wise similarity function is proposed to address the aforementioned non-linear interactions.

B. Heterogeneous Hyper-Network Embedding (HHNE)

The proposed HHNE comprises of two modules: embedding and similarity function learning. In this section, each module and the loss function are explained in details.

1) Embedding Learning: The embedding layer comprises of a fully connected layer with non-linear activation and a two-layer spatial GCN. GCNs can be considered as a generalization of CNNs for graph structured data, where the graph structure is embedded into node-level representations [19]. The spatial GCN is based on weight sharing principal among the local neighborhoods in the graph, where multiple layers of the GCN architecture considers high-order neighborhoods. In this study, node embeddings of a hyperedge are expected to be similar in a tuple-wise manner. In addition, the discrepancy between the embeddings of the neighboring nodes should be small, which can be achieved by GCN. Spatial GCN requires initial node representations and normalized adjacency matrix as given in Eq. 1 [19].

$$\begin{aligned} \mathbf{Z}_1 &= f\left(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(1)}_{gcn}\right) \\ \mathbf{Z}_2 &= f\left(\hat{\mathbf{A}}\mathbf{Z}_1\mathbf{W}^{(2)}_{gcn}\right) \end{aligned} \tag{1}$$

where $\hat{\mathbf{A}} = \mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is normalized adjacency matrix, $\mathbf{D}$ is the diagonal degree matrix, $\mathbf{X}$ is original node

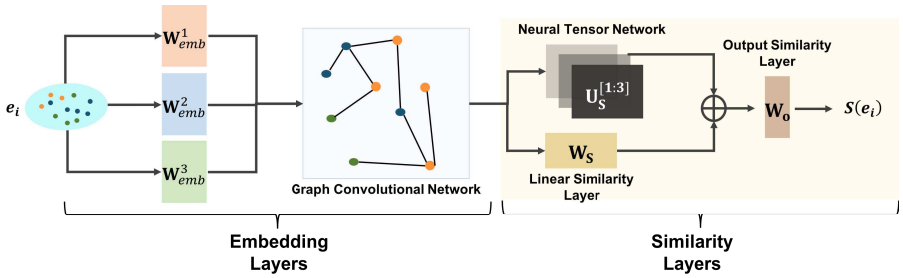

Figure 1: Proposed framework for 3 types of nodes. In the embedding layer, a fully connected layer projects different types of nodes into a common space and 2-layer GCN produces the final embeddings. The similarity layer computes the tuple-wise similarity in the embedding space.

representations, $\{\mathbf{W}^{(1)}_{gcn}, \mathbf{W}^{(2)}_{gcn}\}$ are spatial GCN weights, and $f(\cdot)$ is a non-linear function, such as $\tanh$. Adjacency matrix of a hyper-network is computed as $\mathbf{A} = \mathbf{H\Phi H}^T - \mathbf{D}$ [20], where $\mathbf{H} \in \mathbb{R}^{|V|\times|E|}$ and $h(v,e) = 1$ if node $v \in e$, $\mathbf{\Phi}$ is a diagonal matrix of hyper edge weights $w(e)$ (equals to identity matrix in this study), and $\mathbf{D}$ comprises of $d_v = \sum_{e\in E} w(e)\, h(v,e)$.

GCN approach described above requires same dimensional node representations. However, heterogeneous hyper-network has nodes with different initial dimensionalities, but with semantic relationships. As in a patient-drug-adverse reaction network, it is not possible to model a direct link between patient, drug, and disease attributes. For this reason, a fully connected layer is used to project different types of nodes into a common space before the GCN layer. The first layer embedding, $\mathbf{x}^k_i$, of the ith node of type k, $\mathbf{v}^k_I$, is computed as $\mathbf{v}^k_i = f\left(\mathbf{W}^k_{full}\mathbf{v}^k_i + \mathbf{b}^k_{full}\right)$, where $f(\cdot)$ is a non-linear activation function, $\mathbf{W}^k_{full} \in \mathbb{R}^{r^{'}\times d_k}$ and $\mathbf{b}_{full} \in \mathbb{R}^{r^{'}}$ are the weight and bias of the fully connected layer, and d_k and r are the dimensionalities of type k and the projection space, respectively. Different weights are used for different node types. After mapping the nodes to an $r^{'}$ dimensional space, graph signal $\mathbf{X}$ can be formed with the first layer node embeddings. The output of the GCN layer produces the final embeddings.

2) Similarity Function: The tuple-wise similarity score of the learned node embeddings in a hyperedge is desired to be higher compared with unknown nodes to preserve the hyper-network structure. To ensure the learned embeddings satisfy this hypothesis, a similarity function is simultaneously learned with the embedding layer. Bilinear similarities between cross-type nodes are computed by neural tensor network (NTN) idea proposed in [21]. NTN has bilinear and linear tensor layers to model the relationship between two entities across multiple dimensions. The NTN model computes a similarity score that represents the likelihood of two entities being in a certain type of relationship [21].

The proposed similarity function, given in Eq. 2, adopts the bilinear neural tensor, where the number of dimensions equal to the number of different cross-type interactions in hyperedges. Unlike NTN, the proposed model considers the composite relationship between all the nodes in the hyperedge rather than enforcing the pairwise nodes belonging to an event.

$$S(e) = f\left(\mathbf{h}^{[1:L]} + \sum_{\mathbf{x}_i\in e}\mathbf{W}_S\mathbf{x}_i + \mathbf{b}_S\right)^T \mathbf{W}_o$$
$$\mathbf{h}^{\ell} = \sum_{\mathbf{x}_i,\mathbf{x}_j\in e}(\mathbf{x}_i)^T\mathbf{U}^{\ell}\mathbf{x}_j \quad (2)$$

where f is a non-linear activation function, e is a hyperedge, $\mathbf{U}^{\ell}_S \in \mathbb{R}^{r\times r}$, $\ell \in \{1,\cdots,L\}$ $\mathbf{b}_S \in \mathbb{R}^L$, $\mathbf{W}_S \in \mathbb{R}^{L\times r}$. $\mathbf{h}^{[1:L]}$ is an L dimensional vector, in which each element represents the bilinear similarity between cross-type nodes, and $\{\mathbf{x}_i, \mathbf{x}_j\}$ are the learned node embeddings. For instance, if there are 3 different node types, $\{1,2,3\}$, in a hyperedge, the bilinear similarities are computed between $1-2$, $1-3$, and $2-3$ with $\mathbf{U}^{[1:3]}_S$ resulting in a 3 dimensional tensor $\mathbf{h}^{[1:3]}$. Thus, each slice of the neural tensor represents a different type of interaction. To capture linear contributions of different types, a linear layer is added to unify the embeddings of all the types in the hyperedge. In the last layer, similarity scores of L dimensions are combined by $\mathbf{W}_o \in \mathbb{R}^{L\times 1}$ such that the final similarity score is a linear combination of the contributions of different types of relationships. The overall architecture is illustrated in Figure 1.

3) Loss Function: Hyperedges in information networks usually represent the known interactions between the nodes. On the other hand, unknown interactions cannot be simply considered as negative samples since some of them might not be discovered yet. For this reason, a strict binary classification scheme, which relies on randomly sampled negative hyperedges as in contrastive learning [22], can be problematic for network embedding since the unknown nodes may, in fact, form hyperedges and belong to existing ones. In the proposed framework, unknown edges denoted by E_0 are still sampled, however the loss function is based on a ranking rather than classification. The loss function, given in Eq. 3, aims to maximize the number of tuple-wise hyperedge similarity scores that are higher than the average tuple-wise similarity score of the unknown nodes [23].

$$L = \frac{1}{|E|}\sum\nolimits_{e_i\in|E|}\ell\left(\frac{1}{|E_0|}\sum_{e_{0k}\in|E_0|}S(e_{0k}) - S(e_i)\right) \quad (3)$$

where $S(e_i)$ is the tuple-wise similarity score of hyperedge e_i defined in Eq. 2. Similarly, $S(e_{0k})$ is the similarity score of unknown hyperedge and ℓ can be chosen as a non-decreasing function such as the logistic loss, $\ell(x) = \log(1 + \exp(x))$. Back-propagation algorithm with mini-batch stochastic gradient descent framework, where equal number of known and unknown edges are sampled in each batch, is utilized to learn the embedding and similarity layers. The sampling strategy for unknown nodes is presented in the next section.

4) Unknown Hyperedge Sampling Strategy: In the hyper-network, an unknown group of nodes for each known hyperedge is sampled via the following strategy: First, for each hyperedge, the nodes that do not belong to any hyperedge together with the nodes of the corresponding hyperedge are obtained. Then an unknown hyperedge is formed, including an identifier node from the original hyperedge, and random sub-samples of the unrelated nodes of the remaining types. The number of nodes in unknown hyperedges is chosen the same as their known counterparts to avoid a possible bias that might caused by imbalance in hyperedge sizes.

IV. Experiments

Performance of the proposed approach was compared with popular network embedding approaches on two real-world datasets. The proposed model is implemented in Python using Tensorflow [1]. Hidden dimesionalities of GCN embedding layers were set to 128 and 64, and the embedding dimensionality was set to 32 throughout the experiments.

A. Synthetic Experiment

To simulate a heterogeneous attributed hyper-network, three different datasets of size $100, 350, 300$ were randomly generated from normal distribution with means 1.5, 0, 2.5 and variances 1.8, 0.4, 3.6, respectively. Same dimensionality, 256, was chosen for the visualization of original node features of different types, while the embedding dimensionality was set to 32. A hyperedge was formed by randomly choosing one node from the dataset sized 100 and several nodes from the remaining types. Since the hyper-network was constructed randomly, hyperedges could share nodes and some of the nodes might be isolated. Unknown hyperedges were generated using the strategy discussed in Section III-B4.

Original nodes and the nodes after HHNE are visualized using t-Stochastic Neighbor Embedding (t-SNE) [24]. t-SNE is a dimensionality reduction and visualization technique that captures the Euclidean distances between data points. Since HHNE is designed to increase the tuple-wise similarity of hyperedge embeddings, t-SNE is expected to map the hyperedge nodes closer to each other compared with unrelated nodes. Result for a hyperedge and corresponding unknown counterpart in Figure 2 shows that the HHNE approach satisfies the hypothesis such that cross-type nodes in a hyperedge get closer after HHNE and the embeddings of the unrelated nodes remain distant to each other.

[1] https://github.com/illidanlab/HHNE.git

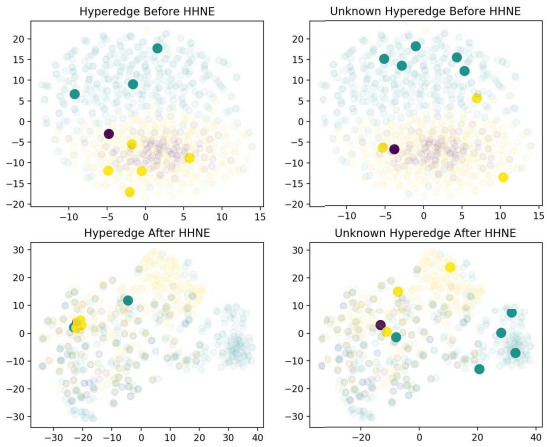

Figure 2: A synthetic hyperedge and unknown hyperedge nodes before and after HHNE. Hyperedges are highlighted by the darker colors and the rest of the nodes can be seen in the background. After HHNE, hyperedge nodes get closer, while the unrelated nodes are still distant to each other.

B. Baselines

In this study, we compare the proposed approach with popular network embedding approaches such as DeepWalk [5], node2vec [25], LINE [4], DHNE [18], and HIN2Vec [17]. We report the performance of first order LINE-1, second order LINE-2, and LINE-1-2, which is a concatenation of LINE-1 and LINE-2. LINE-1 considers local pairwise proximities, whereas LINE-2 considers second order neighborhood. To convert hyper-network to a standard graph, the pairwise links in hyperedges were considered. For HIN2Vec [17], different types of edges were formed, such as T1-T2, T1-T3, T2-T3, T1-T2-T3, when there are three types of nodes as T1,T2, and T3. For DHNE [18], hyperedges were decomposed into hyperedges of size 3. Baseline configurations suggested by the authors were used, except the embedding dimension was set to 32. The baseline algorithms do not provide a mechanism to compute the tuple-wise similarities of the hyperedges. For this reason, the tuple-wise similarities are compared using the equation below:

$$S(e) = \frac{1}{|e|} \sum_{\{\mathbf{x}_i, \mathbf{x}_j\} \in e, i \neq j} \mathbf{x}_i^T \mathbf{x}_j \tag{4}$$

where $|e|$ is the number of nodes in hyperedge e and $\mathbf{x}_i$ denotes the normalized learned embedding.

C. Real-World Datasets

In this study, two real-world datasets were used to evaluate the HHNE performance.

1) Drug-Drug Interactions (DDI) Dataset: For evaluation of HHNE performance on real-world tasks, a DDI dataset is used with 526 drugs and the following views: (1) **Drug Indication**: $1,702$ conditions that the drugs cure were downloaded from SIDER database [2]. (2) **Chemical Structure**: 582 dimensional drug features were extracted from an open chemisty database, PubChem [3]. (3) **Single Drug Adverse Reaction**: 327 high low level hierarchy Adverse Drug Reactions (ADRs) were extracted from ADReCS [4]. In summary, a drug hyper-network was manually constructed with $2,555$ nodes and 3 types. For drug indication and ADR, one-hot representations were used as the initial node features. For every drug, a hyperedge was formed including its chemical structure, several indications and ADRs. DDI network had, in total, 526 hyperedges with an average of 79 nodes, where every node belongs at least one hyperedge.

2) FAERS Dataset: The FDA Adverse Event Reporting System (FAERS) is a self-reported adverse event and associated medications database [5]. In the experiments, the cleansed version of 2016 FAERS data [26] was used. Each report comprises of a patient, several drugs and their ADRs. After eliminating patients with missing values, $32,955$ hyperedges were formed including a patient, drugs and ADRs he experienced. Age group, gender, weight and report type were used as patient node features, where the categorical ones were coded as one-hot vectors. In total, there were $32,955$ patient, $20,154$ drug and $6,605$ ADR nodes in the network.

D. Hyperedge Detection

In this experiment, the performance of the proposed hyper-network embedding approach is evaluated for hyperedge detection. In this task, which can also be considered as link prediction, 20% of the known hyperedges were hidden during the training. 80% of the known edges and randomly generated unknown hyperedges were used to train the proposed model. Test similarity scores were sorted in decreasing order to compute the average precision (AP) and precision@k (Prec@k), where k equals to the number of known hyperedges. Tuple-wise similarities of baselines were computed by Eq. 4. HHNE performance for DDI and FAERS data is summarized in Table I. Experimental results indicate that simultaneously learning the embeddings with the similarity function can improve the tuple-wise similarity scores. DeepWalk also performed well for DDI dataset. HHNE-2 row in the table reports the performance of the embeddings learned by the proposed HHNE with the similarity function in Eq. 4 without using the learned similarity function. Baseline performance for FAERS dataset were very poor compared with HHNE. This result shows that learning a similarity function can be relevant in such cases.

[2]http://sideeffects.embl.de/

[3]https://pubchem.ncbi.nlm.nih.gov/

[4]http://bioinf.xmu.edu.cn/ADReCS/

[5]https://open.fda.gov/data/faers/

Table I: Hyperedge detection performance for DDI and FAERS datasets. Prec@k stands for precision@k where k equals to the number of hyperedges and AP is the average precision. Average performance of 5 random splits are shown for HHNE. HHNE-2 denotes computing the tuple-wise similarity using Eq. 4 after embedding learning by HHNE.

	DDI		FAERS	
Method	Prec@k	AP	Prec@k	AP
HHNE	0.98	0.99	0.95	0.98
HHNE-2	0.86	0.94	0.89	0.96
HIN2Vec	0.57	0.66	0.47	0.48
DHNE	0.89	0.96	0.25	0.35
DeepWalk	1.00	1.00	0.28	0.34
node2vec	0.58	0.63	0.68	0.68
LINE-1	0.84	0.94	0.55	0.55
LINE-2	0.59	0.66	0.62	0.63
LINE-1-2	0.63	0.73	0.57	0.56

Table II: DDI detection performance. AUC and F1-score are computed for the ranked similarity of the positive and negative DDI pairs.

Methods	AUC	F1-Score
HHNE	0.64	0.61
HIN2Vec	0.69	0.64
DHNE	0.60	0.56
DeepWalk	0.55	0.50
node2vec	0.56	0.52
LINE-1	0.53	0.48
LINE-2	0.59	0.55
LINE-1-2	0.61	0.57

On the other hand, HHNE-2 could still outperform baselines when Eq. 4 was used.

1) DDI Detection: DDI detection is a crucial step to prevent mortalities and injuries caused by adverse reactions due to DDI. In the DDI dataset, 222 drugs out of 526, resulting in $8,576$ pairs, have adverse interactions. In addition to known DDI pairs, $17,152$ unknown pairs were sampled from the rest of the drugs. After the embedding learning, pairwise similarities between known and unknown pairs are computed and ranked in a decreasing order. Top $8,576$ similarity scores are assumed to be predicted as the DDI pairs. Based on this assumption, AUC and F1 scores in Table I are computed. Overall performance is poor, however HHNE and HIN2Vec produced the best results among other baselines. This result indicates that considering composite relationships can be more helpful for inference from heterogeneous neighboring relationships.

V. Conclusion

In this study, a deep embedding learning approach is proposed for heterogeneous hyper-networks, where hyperedges contain several different types of nodes. HHNE simultaneously learns an embedding and a tuple-wise similarity

function to preserve hyperedges in the embedding space. The proposed fully-connected and GCN layers capture non-linear interactions between nodes and project them into a common space. The similarity function defines the tuple-wise similarities without decomposing them into pair-wise predefined interactions. Experiments on real-world datasets showed that HHNE preserves the heterogeneous composite interactions better than the baseline methods. In the future work, model complexity will be investigated.

ACKNOWLEDGMENT

This research is supported in part by the Office of Naval Research (ONR) under grants number N00014-17-1-2265 (to JZ and AKJ), N00014-14-1-0631 (to JZ and AKJ) and National Science Foundation under grants IIS-1565596, IIS-1615597, and IIS-1749940 (to JZ). This research of Fei Wang is supported by ONR N00014-18-1-2585, NSF IIS-1716432 and NSF IIS-1750326.

REFERENCES

[1] 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.

[2] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, L. Kaplan, and J. Han, "Embedding learning with events in heterogeneous information networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 11, pp. 2428–2441, 2017.

[3] Y. Zhu, Z. Guan, S. Tan, H. Liu, D. Cai, and X. He, "Heterogeneous hypergraph embedding for document recommendation," *Neurocomputing*, vol. 216, pp. 150 – 162, 2016.

[4] 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.

[5] 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.

[6] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI 17*, 2017.

[7] J. Liang, P. Jacobs, and S. Parthasarathy, "SEANO: semi-supervised embedding in attributed networks with outliers," *CoRR*, vol. abs/1703.08100, 2017. [Online]. Available: http://arxiv.org/abs/1703.08100

[8] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, 2016, pp. 1145–1152.

[9] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. JMLR.org, 2016, pp. 2014–2023.

[10] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1225–1234.

[11] L. Liao, X. He, H. Zhang, and T.-S. Chua, "Attributed social network embedding," *CoRR*, vol. abs/1705.04969, 2017. [Online]. Available: https://arxiv.org/abs/1705.04969

[12] R. A. Rossi, R. Zhou, and N. K. Ahmed, "Deep feature learning for graphs," *CoRR*, vol. abs/1704.08829, 2017. [Online]. Available: https://arxiv.org/abs/1704.08829

[13] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *CoRR*, vol. abs/1706.02216, 2017. [Online]. Available: http://arxiv.org/abs/1706.02216

[14] J. Tang, M. Qu, and Q. Mei, "PTE: predictive text embedding through large-scale heterogeneous text networks," *CoRR*, vol. abs/1508.00200, 2015. [Online]. Available: http://arxiv.org/abs/1508.00200

[15] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *CoRR*, vol. abs/1610.09769, 2016. [Online]. Available: http://arxiv.org/abs/1610.09769

[16] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD '17*. ACM, 2017, pp. 135–144.

[17] 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.

[18] K. Tu, P. Cui, X. Wang, F. Wang, and W. Zhu, "Structural deep embedding for hyper-networks," *CoRR*, vol. arXiv preprint arXiv:1711.10146, 2017. [Online]. Available: https://arxiv.org/pdf/1711.10146

[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations*, 2017.

[20] F. Wu, Y.-H. Han, and Y.-T. Zhuang, "Multiple hypergraph clustering of web images by miningword2image correlations," *Journal of Computer Science and Technology*, vol. 25, no. 4, pp. 750–760, Jul 2010.

[21] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 2013, pp. 926–934.

[22] N. A. Smith and J. Eisner, "Contrastive estimation: Training log-linear models on unlabeled data," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005, pp. 354–362.

[23] Y. Liu, S. Qiu, P. Zhang, P. Gong, F. Wang, G. Xue, and J. Ye, "Computational drug discovery with dyadic positive-unlabeled learning," in *Proceedings of the 2017 SIAM International Conference on Data Mining*, 2017, pp. 45–53.

[24] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2008, pp. 2579–2605, 2008.

[25] 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.

[26] C. I. L. at National University of Kaohsiung. (2016) Interactive adverse drug reactions system. [Online]. Available: http://iadr.csie.nuk.edu.tw/Resource.aspx