HINGCN: Heterogeneous information network revisited with GCN

Danhao Ding*, Xiang Li*, Nikos Mamoulis†, Ben Kao*
*The University of Hong Kong, Pokfulam Road, Hong Kong
†University of Ioannina, Ioannina, Greece
*{dhding2, xli2, kao}@cs.hku.hk †nikos@uoi.gr

ABSTRACT

GCN on HIN.

KEYWORDS

Semi-supervised classification; graph convolution; heterogeneous information network

ACM Reference Format:
Danhao Ding*, Xiang Li*, Nikos Mamoulis†, Ben Kao*. 2018. HINGCN: Heterogeneous information network revisited with GCN. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3178876.3185993

1 INTRODUCTION

Dan:TODO

Our main contributions are:

• We proposed a heterogeneous graph convolution algorithm that is capable of capturing edge information.
• Dan:don't know
• We conduct extensive experiments to evaluate the performance of HINGCN against 9 other classification methods. Our results show that HINGCN performs very well against the competitors. In particular, it is very robust in that it consistently performs well over all the datasets tested. Also, it outperforms others by wide margins for datasets that are highly multi-scale.

The rest of the paper is organized as follows. Section 2 mentions related works on heterogeneous graph neural networks, graph embedding and described several semi-supervised classification algorithms. Section 4 presents the HINGCN algorithm. Section 5 describes the experiments and presents experimental results. Finally, Section 6 concludes the paper.

2 RELATED WORK

2.1 Heterogeneous graph neural networks

Dan:TODO

2.2 Heterogeneous graph embedding

Dan:TODO

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5639-8/18/04..
https://doi.org/10.1145/3178876.3185993

Table 1: Description of symbols

Notation	Description
Φ	Meta-path
N^{Φ}	Set of neighbors with respect to a given meta-path Φ
$h^{(t)}$	Node embedding at layer t; h^0 is input node feature
$r^{(t)}$	Edge embedding at layer t; r^0 is input edge feature
$e^{\Phi}_{i,j}$	Similarity of node pair (i,j) under meta-path Φ
$a^{\Phi}_{i,j}$	Attention of node pair (i,j) under meta-path Φ
W_F	Weight matrix of fully connected layer F
σ	Sigmoid function
z_i	Output embedding for node i

3 PRELIMINARY

Definition 1. **Heterogeneous Information Networks (HIN)**[1]. Let $T = \{T_1, ..., T_m\}$ be a set of m object types. For each type T_i, let n_i and $X_i = x_i 1, ..., x_{in_i}$ be the number and the set of objects of type Ti, respectively. An HIN is a graph $G = (V, E)$, where $V = \bigcup_{i=1}^{m} X_i$, and E is a set of links, each represents a binary relation between two objects in V. If $m = 1$ (i.e., there is only one object type), G reduces to a homogeneous information network.

Example 1. *Dan:TODO*

Definition 2. **Metapath**[2]. A meta-path Φ is a path defined on a graph schema. Meta-path Φ: $T_1 \xrightarrow{R_1} \cdots \xrightarrow{R_l} T_{l+1}$ defines a composite relation $R = R_1 \circ \cdots \circ R_l$ that relates objects of type T_1 to objects of type T_{l+1}. We say a path $p = (x_1 x_2 \ldots x_{l+1})$ between x_1 and x_{l+1} in network G follows the meta-path Φ, if each object is of type T_i and each link $ei = \langle x_i x_{i+1} \rangle$ belongs to each relation R_i in Φ. We say that $p_{x_1 \rightsquigarrow x_{l+1}}$ is an instance of meta-path Φ, denoted by $p_{x_1 \rightsquigarrow x_{l+1}} \vdash \Phi$. Moreover, in this paper we say x_v is a meta-path Φ related neighbor of x_u, denoted by $x_v \in N^{\Phi}(x_u)$

Example 2. *In Fig.* **??**, *Dan:TODO*

The notations throughout the rest of paper are shown in Table 1.

4 ALGORITHM

In this section we describe our HINGCN algorithm. Figure **??**(c) shows a flow diagram of HINGCN. Dan:TODO:signposting

4.1 Edge features as supplementary information

We claim that information is loss if we only consider adjacency of target type vertices. For example, in long tailed meta-path *APCPA*,

a shrunk *A-A* adjacency loses information from intermediate *PCP* nodes and cannot distinguish between two instance $p_1 = (a_1 p_1 c_1 p_2 a_2)$ and $p_2 = (a_1 p_3 c_2 p_4 a_2)$. Therefore we propose to add this information back by adding additional edge feature to homogeneous *A-A* edges. These edge features are composed of three different components, namely intermediate node embeddings, path count and PathSim[2].

•**Intermediate node embedding**.

•**Path Count**. Different object pairs have different number of path instance between them and this number may reflect object similarity under a given meta-path. We propose to record this count of path between objects x_u and x_v under meta-path Φ by:

$$c(x_u, x_v) = |\{p_{x_u \rightsquigarrow x_v} : p_{x_u \rightsquigarrow x_v} \vdash \Phi\}|$$

•**PathSim**. PathSim is proposed in [2] to measure similarity between two objects of a same type on a heterogeneous graph. Given a symmetric meta-path Φ, PathSim between x_u and x_v is:

$$s(x_u, x_v) = \frac{2 \times |\{p_{x_u \rightsquigarrow x_v} : p_{x_u \rightsquigarrow x_v} \vdash \Phi\}|}{|\{p_{x_u \rightsquigarrow x_u} : p_{x_u \rightsquigarrow x_u} \vdash \Phi\}| + |\{p_{x_v \rightsquigarrow x_v} : p_{x_v \rightsquigarrow x_v} \vdash \Phi\}|}$$

Dan:TODO

4.2 Aggregating node embedding

Dan:TODO

$$X = XZ. \tag{1}$$

4.3 Aggregating edge embedding

Dan:TODO

$$X = XZ. \tag{2}$$

4.4 Dealing with over-smoothing via adaptive depth

Dan:TODO Inspired by DenseNet and ResNet, we alleviate the problem of over-smoothing over graph convolutions by creating jumping connections.

$$X = XZ. \tag{3}$$

4.5 Aggregation across different meta-path semantics

Dan:TODO

$$X = XZ. \tag{4}$$

Dan:need some intuition/justification shit Given a node i, and its set of embeddings under different semantics, a metapath aggregator is a function γ in the form of $y_i = \gamma_\Theta(\{x_{i,1}, x_{i,2}, ..x_{i,P}\})$, where $x_{i,p}$ is the embedding of node i under metapath p, and y_i is the aggregated embedding of node i in the HIN. Θ is the learnable paremeters of the aggregator. Here we propose several metapath aggregators.

• **Attention**

• **Gated linear unit** Inspired by forget gate mechanisms in recurrent networks (RNN) Dan:TODO:cite, we compute a soft gate between 0 (low importance) and 1 (high importance) to represent different importance to each embedding entry. We make the element-wise gated sum layer as follows:

$$o_{\Phi_p} = \sigma(W_o^T Z_{\Phi_p}), \qquad \tilde{Z}_{\Phi_p} = \tanh(W_z^T Z_{\Phi_p})$$

and

$$Z'_{\Phi_p} = o_{\Phi_p} \odot \tilde{Z}_{\Phi_p} \tag{5}$$

The gated unit o_{Φ_p} with sigmoid output (ranged (0,1)) is used to control which elements are to be aggregated in the output. • **Other aggregators** In preliminary experiments, CONCAT, MAX-POOLING and MEAN aggregators did not generate a better result, and we omit results of these aggregators in experiment section.

4.6 HINGCN: Semi-supervised classification on Heterogeneous Information Network via Graph Convolution Networks

After meta-path aggregation layer, the embeddings are fed to a 2-layer MLP for final output. In semi-supervised classification tasks, we minimize the Cross-Entropy loss and the model generates final prediction based on the largest output value. Finally, HINGCN is summarized in Algorithm 1.

To this end, we summarize ROSC as follows. ROSC first computes the TKNN graph $\mathcal{G}$ and the associated weight matrix $\mathcal{W}_K$. Then it applies power iteration to generate p pseudo-eigenvectors that form a $p \times n$ matrix X. After whitening and normalization on X, the coefficient matrix Z^* can be calculated by solving Eq. **??**, which further leads to the construction of a rectified similarity matrix $\tilde{Z}$. Since $\tilde{Z}$ has the grouping effect, the standard spectral clustering methods (e.g., NCuts) will be finally applied to derive more robust clustering results. The time complexity of ROSC will be no more than the standard spectral clustering methods, which is $O(n^3)$ in general. In the future, we will attempt to improve it.

Algorithm 1 HINGCN

Input: S, k.
Output: $C = \{C_1, \ldots, C_k\}$
1: Compute the TKNN graph and the reachability matrix $\mathcal{W}$
2: Calculate $W = D^{-1}S$, where $D_{ii} = \sum_j S_{ij}$
3: Apply PI on W and generate p pseudo-eigenvectors $\{\boldsymbol{v}_r\}_{r=1}^{p}$
4: $X = \{\boldsymbol{v}_1^T; \boldsymbol{v}_2^T; \ldots; \boldsymbol{v}_p^T\}$; X = whiten(X)
5: Normalize each column vector $\boldsymbol{x}$ of X such that $\boldsymbol{x}^T\boldsymbol{x} = 1$
6: Calculate the coefficient matrix Z^* by Eq. **??**
7: Construct $\tilde{Z} = (|Z^*| + |(Z^*)^T|)/2$
8: Run standard spectral clustering methods, e.g., NCuts, with $\tilde{Z}$ as the similarity matrix to obtain clusters $C = \{C_r\}_{r=1}^{k}$
9: **return** $C = \{C_1, \ldots, C_k\}$

5 EXPERIMENT

We conducted extensive experiments to evaluate the performance of HINGCN. This section summarizes our results. We compare the various methods using three popular measures, namely, *accuracy*, *micro-F1*, and *macro-F1*. These measures evaluate clustering quality and their values range from 0 to 1, with a larger value indicating a better clustering quality.

5.1 Datasets

A summary of statistics of the datasets rae shown in table **??**.

• **DBLP**: We extract a subset of DBLP with contains 4057 authors (*A*), 14328 papers (*P*) and 20 conferences (*C*). 8789 terms of papers are aggregated to each author as feature, after tf-idf transformation. Here we select meta-paths {*APA*, *APAPA*, *APCPA*} for experiments.
• **Yelp**: A standard spectral clustering method with symmetric normalization.
• **Freebase**: A standard spectral clustering method with symmetric normalization.

5.2 Algorithms for comparison

We evaluate HINGCN and 9 other methods. To demonstrate effectiveness of our edge update mechanism and meta-path GLU layer, we also includes two variants of HINGCN.

• **Node2vec**: A standard spectral clustering method with symmetric normalization.
• **Metpath2vec**: A standard spectral clustering method with divisive normalization.
• **GCN**: A self-tuning spectral clustering method for multi-scale clusters. It uses eigenvector rotation to estimate the number of clusters. To make a fair comparison, we directly set k as in other methods.
• **GAT**: A power iteration based method which generates only one pseudo-eigenvector.
• **HAN**: A hierarchical semi-supervised graph attention network that employs node-level attention and meta-path level attention.
• **HINGCN**$_e d$: A variant of HINGCN which replace update of edge embedding by identity propagation.
• **HINGCN**$_a t$: A variant of HINGCN which uses attention mechanism for meta-path level aggregation.
• **HINGCN**: The proposed semi-supervised classification method based on heterogeneous graph convolution networks.

5.3 Experiment setup

Dan:TODO We implement HINGCN using PyTorch. We use the same learning rate 0.001. The models are trained with two NVIDIA 1080Ti GPUs using data parallelism. The batch size is 1024 for each GPU. We stack two graph aggregators, and each aggregator samples 16 neighbors for both training and testing. Each aggregator is then followed by a ReLU activation layer and a dropout layer with dropout rate 0.5. Output dimension of the layers are set to 64 and 32 respectively. For dataset without node features, we encode a 1-hot matrix as input feature. As mentioned in section Dan:TODO, we ensemble edge feature from pre-trained path-sim and node embeddings. These node embeddings are trained using Node2vec with default parameters on the underlying graph neglecting heterogeneity.

The parameters of all the baseline methods are set according to their original papers. For each method and dataset, we run the experiment 10 times and report average results.

5.4 Performance results

We apply all 10 methods on DBLP. Due to space limitations, for each category of methods, we only show the best performing one. They are, namely, NJW, PIC-k, FUSE, and ROSC.

6 CONCLUSIONS

In this paper we studied the effectiveness of spectral clustering methods in handling multi-scale data. We discussed the traditional approaches of locally scaling the similarity matrix and power-iteration-based techniques. We described the methods ZP and FUSE, which were previously proposed to cluster multi-scale data. We pointed out that these existing approaches focus on measuring the correlations of objects via feature similarity. However, for data with various sizes and densities, objects that belong to the same big cluster could be at substantial distances from each other. Feature similarity could fail in this case. In view of this, we proposed ROSC, which computes an affinity matrix $\tilde{Z}$ that takes into account both feature similarity and reachability similarity. In particular, $\tilde{Z}$ is obtained by regularizing a primitive affinity matrix with a TKNN graph. We mathematically proved that $\tilde{Z}$ has the desired grouping effect, which makes it a very effective replacement of the similarity matrix S used in spectral clustering. We conducted extensive experiments comparing the performance of ROSC against 9 other methods. Our results show that ROSC provides very good performances across all the datasets, both real and synthetic, we tested. In particular, for cases where the datasets are highly multi-scale, ROSC substantially outperforms other competitors. ROSC is therefore a very robust solution for clustering multi-scale data.

7 ACKNOWLEDGMENTS

This research is supported by Hong Kong Research Grants Council GRF HKU 17254016.

REFERENCES

[1] Ming Ji, Yizhou Sun, Marina Danilevsky, Jiawei Han, and Jing Gao. 2010. Graph Regularized Transductive Classification on Heterogeneous Information Networks. In *ECML/PKDD (1) (Lecture Notes in Computer Science)*, Vol. 6321. Springer, 570–586.

[2] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *PVLDB* 4, 11 (2011), 992–1003.