Attention Based Meta Path Fusion for Heterogeneous Information Network Embedding

Houye Ji, Chuan Shi(✉), and Bai Wang

Beijing University of Posts and Telecommunications, Beijing, China
{jhy1993,shichuan,wangbai}@bupt.edu.cn

Abstract. Recently, there is a surge of network embedding algorithms, which embed information network into a low dimensional space. However, contemporary network embedding algorithms focus on homogeneous networks, while we know that many real-world systems can be constructed with heterogeneous information networks (HINs). Compare to homogeneous networks, HINs contain heterogeneity types of nodes and edges, which leads to new challenges for traditional network embedding: handing mixed heterogeneous nodes and fusing rich semantic information. Although several HIN embedding algorithms have been proposed, these challenges have not been well dressed. How to explore the rich semantic information and integrate these information still remain to be solved. In this paper, we propose a novel attention based meta path fusion model for HIN embedding (called AMPE). In order to handle node heterogeneity and extract rich information, AMPE first extracts multiple homogeneous networks from HIN with meta paths, and then employs adopted AutoEncoders to embed these homogeneous networks. After that, AMPE fuses these embeddings learned from homogeneous networks with attention mechanism. Experimental results on two real-world datasets demonstrate the effectiveness of the proposed model.

1 Introduction

Recent years, network representation learning [1–3] (e.g. network embedding) has attracted a great deal of attention. The goal of network representation learning is to embed a network into a low dimensional latent space, in which each node is represented as a latent vector. Such representations can preserve the proximities between the nodes, which can be treated as feature vectors and applied in subsequent data mining tasks, such as node classification, community detection and link prediction.

However, most network embedding algorithms focus on homogeneous network containing the same types of node or edge. In the real world, networks usually contain multiple types of nodes or edges, named heterogeneous information networks (HINs). Since HINs can model much more complex relationships and structures than homogeneous information networks, it has been widely used in graph mining [4,5]. Meanwhile, meta path [4], a relation composition connecting two types of nodes, has been widely used to capture rich semantic information contained in HIN. Taking Fig. 1(b) as an example, a meta path between two businesses can be Business-City-Business, which means these businesses located in the same city. Due to the complexity of HIN, traditional homogeneous network embedding methods cannot be directly applied to HIN because of the following two reasons:

(1) Heterogeneous information networks have various types of nodes and edges. How to preserve the heterogeneous neighbors of each node is an urgent problem that need to be solved. (2) Heterogeneous information networks contain rich and complex semantic information. How to extract and fuse these information is an open problem. A novel fusion model that can select some useful information and combine them will be desired.

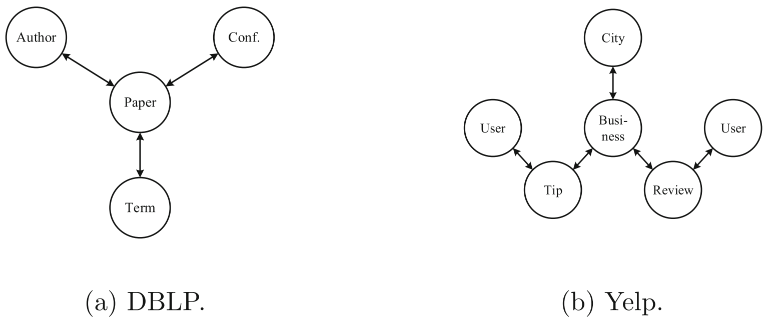


Fig. 1. Two examples of heterogeneous information network in experiments.

Some HIN embedding algorithms have been proposed to overcome above challenges. Dong et al. [6] introduce meta path based random walk to embed heterogeneous network, which only utilizes single meta path to extract semantic information. HIN2Vec [7] is designed to capture the semantic embedded by exploiting different meta paths among nodes. Esim [8] tries to capture semantic information from multiple meta paths. However, it relies heavily on user-specified meta path and adopts grid search to obtain the weights of meta paths. The network embedding algorithms aforementioned usually embed the semantic information via random walk model, which maybe unable to fully capture the deep semantic information. What's more, since HINs contain rich semantic information, instead of utilizing a single meta path, the combination of multiple meta paths can give a more comprehensive description of HINs.

To better solve challenges faced by HIN embedding, we propose a novel **A**ttention based **M**eta **P**ath fusion heterogeneous network **E**mbedding model, named AMPE, which can embed various kinds of semantic information extracted from multiple meta paths and fuses them for specific tasks. In AMPE, we first transform the original heterogeneous network into several homogeneous networks based on corresponding meta paths. Then AMPE extends AutoEncoder to the heterogeneous scenario that can extensively embed these semantics information simultaneously. Significantly different from current weight learning for meta paths, we propose an attention based deep fusion module, which can weight the importance of each meta path and fuses them for specific tasks.

The remainder of this paper is organized as follows. We first review the related work in Sects. 2 and 3 gives the preliminary concepts in heterogeneous information network. In Sect. 4, we introduce the proposed model in details. Datasets and experiments are presented in Sect. 5. Finally, some conclusions and future works are showed in Sect. 6.

2 Related Work

Originally, network embedding algorithms were proposed to embed network into a low dimensional latent space, in which each node is represented as a latent vector. The motivation behind those algorithms is to preserve the structural information of nodes, so the learned embeddings can be applied to further data mining tasks.

Recently, some network embedding algorithms based on random walk and deep learning have been proposed. Inspired by word embedding [9], DeepWalk [10] embeds network structures by truncated random walk. Along with DeepWalk, node2vec [11] designs a biased random walk to sample the neighbors. Meanwhile, LINE [12] is an efficient network embedding method that can deal with large-scale networks. Wang et al. [13] introduce deep learning to perform network embedding. However, all of these algorithms cannot be applied to HINs directly because they didn't consider the heterogeneity of HINs.

There are also some heterogeneous network embedding algorithms. Esim [8] proposes a meta path guided embedding method to perform similarity search for HINs. Metapath2vec [6] designs a meta path based random walk and apply skip-gram to learn network representations. HIN2Vec [7] captures the rich semantics information in HINs by exploiting different types of relationships among nodes. However, most of existing methods can't learn the importance of meta paths and combine them for specific tasks.

3 Preliminaries

A heterogeneous information network is a special kind of information network, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which consists of an object set $\mathcal{V}$ and a link set $\mathcal{E}$. A HIN is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and a link type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$. $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined object types and link types, where $|\mathcal{A}| + |\mathcal{R}| > 2$.

The complexity of heterogeneous information network drives us to provide the meta level (i.e., schema-level) description for understanding the object types and link types better in the network. Given the typed essence, a HIN can be abstracted as a network schema, denoted as $\mathcal{S} = (\mathcal{A}, \mathcal{R})$ [5,14], which is a meta template define over object types. Figure 1 gives network schemas on both DBLP and Yelp.

A meta path Φ [4] is a path defined on a schema $\mathcal{S} = (\mathcal{A}, \mathcal{R})$, and denoted in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \dots \xrightarrow{R_l} A_{l+1}$. Given a meta path Φ, we can translate the original HIN G into a homogeneous network G_Φ which can capture the semantic information of meta path. The adjacency matrix of G_Φ can be represented as $M_\Phi \in \mathbb{R}^{n \cdot n}$, where $M_\Phi(i,j) = 1$ iff node i connects to node j via meta path Φ.

4 The Proposed Method

In this section, we propose a novel deep model to embed nodes in HIN into low-dimensional vectors, called AMPE. AMPE is composed of three major components. First, we transform the original HIN into several homogeneous networks. After that, AMPE embeds these homogeneous network simultaneously by an adopted AutoEncoders. Finally, AMPE can learn the weight of each meta path based embedding and fuses them via attention mechanism. Figure 2 presents the framework of AMPE.

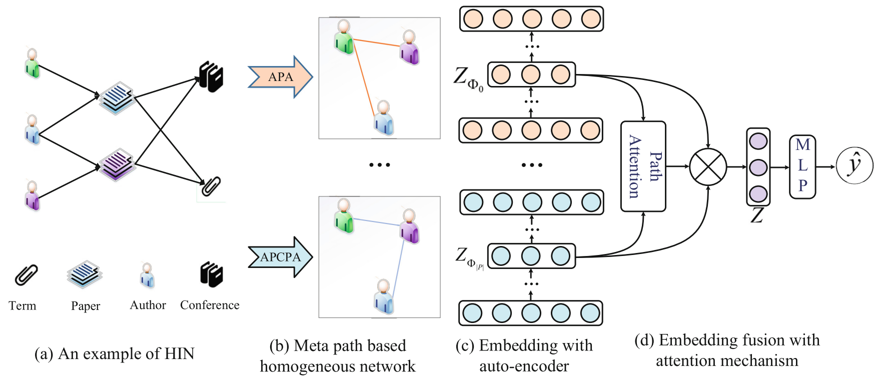

APA
APCPA
Term
Paper
Author
Conference
Z_{Φ_0}
$Z_{\Phi_{|P|}}$
Path Attention
Z
MLP
$\hat{y}$
(a) An example of HIN
(b) Meta path based homogeneous network
(c) Embedding with auto-encoder
(d) Embedding fusion with attention mechanism

Fig. 2. The framework illustration of the proposed AMPE approach.

4.1 Meta Path Based Homogeneous Network

Heterogeneous information networks usually have various kinds of complex semantic information. Here we utilize a set of meta paths (e.g., $\Phi_0, \Phi_1, \dots, \Phi_{|P|}$) to extract semantic information and transform the original HIN G into several homogeneous networks (e.g., $G_{\Phi_0}, G_{\Phi_1}, \dots, G_{\Phi_{|P|}}$). Each homogeneous network G_Φ contains one type of semantic information which means every node is connected to its neighbors through meta path Φ. Here we give the adjacency matrix of each homogeneous network G_Φ, denoted as M_Φ. In homogeneous network G_Φ, if node i is connected to node j through meta path Φ, then $M_\Phi(i,j) = 1$.

4.2 Homogeneous Network Embedding via AutoEncoder

After obtaining several meta path based homogeneous networks, we extend traditional AutoEncoder to embed these homogeneous networks simultaneously.

Here we give a brief review of AutoEncoder. AutoEncoder [15] is an unsupervised deep learning model that can copy its input to its output. It usually involves two parts: encoder, which encodes the original feature representations into a latent space; decoder, which try to reconstruct the original representations from latent space. Formally, let x_i denotes the original feature representation of node i, and $\{y_i^1, y_i^2, \ldots, y_i^k\}$ denote the hidden representation of each encoder layer. The relationships between these representations can be shown as follows:

$$y_i^1 = \sigma(W^1 x_i + b^1), \tag{1}$$

$$y_i^k = \sigma(W^k y^{k-1} + b^k), k = 2, \ldots, K, \tag{2}$$

$$z_i = \sigma(W^{K+1} y_i^k + b^{K+1}). \tag{3}$$

Here z_i means the learned representation and σ is sigmoid function. After obtaining z_i, we try to reconstruct the original representation, denoted as $\hat{x}_i$. In decoder, the latent representations of hidden layers can be denoted as $\{\hat{y}_i^k, \hat{y}_i^{k-1}, \ldots, \hat{y}_i^1\}$ and the relationships between them can be shown as follows:

$$\hat{y}^K = \sigma(\hat{W}^{K+1} z_i + \hat{b}^{K+1}), \tag{4}$$

$$\hat{y}_i^{k-1} = \sigma(\hat{W}^k \hat{y}^k + \hat{b}^k), k = 2, \ldots, K, \tag{5}$$

$$\hat{x}_i = \sigma(\hat{W}^1 \hat{y}_i^1 + \hat{b}^1). \tag{6}$$

The learning process of AutoEncoder is to minimize the distance between the original feature representation x_i and the reconstructed representation $\hat{x}_i$, denoted as follows:

$$L = \sum_{i=1}^{n} ||x_i - \hat{x}_i||_2^2. \tag{7}$$

Unfortunately, such an AutoEncoder cannot directly be applied to network embedding due to the sparsity problem, which means the number of zero elements in adjacency matrix is far more than non-zero elements. In order to solve this problem, we impose more penalty on non-zero elements. Then AutoEncoder will pay more attention to these non-zero elements and gives priority to reconstructing them. The modified loss function can be shown as follows:

$$L = \sum_{i=1}^{n} ||(x_i - \hat{x}_i) \odot b_i||_2^2 = ||(\hat{X} - X) \odot B||_F^2. \tag{8}$$

where $\odot$ means Hadamard product, b_i is a weight vector and $b_{i,j} = x_{i,j} * (\beta - 1) + 1$. Here β is penalty coefficient.

By fitting these AutoEncoders adjacency matrix of each homogeneous network simultaneously, we can obtain a group of node embeddings. The overall loss function for these AutoEncoders can be summarized as below:

$$L_{ae} = \sum_{i=1}^{|P|} ||(\widehat{X_{\Phi_i}} - X_{\Phi_i}) \odot B_{\Phi_i}||_F^2. \tag{9}$$

After training these AutoEncoders, we can obtain $|P|$ groups of node embeddings, denoted as $\left\{Z_{\Phi_0}, Z_{\Phi_1}, \ldots, Z_{\Phi_{|P|}}\right\}$.

4.3 Fusing Embeddings via Attention Mechanism

After obtaining $|P|$ groups of node embeddings, the proposed AMPE can automatically fuse these embeddings and learn their importances via attention mechanism for the specific task. The fusion process F can be shown as follows:

$$Z = F(Z_{\Phi_0}, Z_{\Phi_1}, \ldots, Z_{\Phi_{|P|}}). \tag{10}$$

Inspired by the attention mechanism in neural machine translation [16], we define the attention value as follows:

$$att'_{\Phi_i} = h^T \cdot Tanh(W \cdot Z_{\Phi_i} + b), \tag{11}$$

$$att_{\Phi_i} = \frac{exp(att'_{\Phi_i})}{\sum_{i=1}^{|P|} exp(att'_{\Phi_i})}, \tag{12}$$

where W is a weight matrix, b is a bias vector, h is weight vector, att_{Φ_i} is the weight of meta path Φ_i. Obviously, the higher att_{Φ_i}, the more important meta path Φ_i is. With the learned weights as coefficients, we can weight combine these embeddings to obtain the final embedding. The final embedding Z is shown as follows:

$$Z = \sum_{i=1}^{|P|} att_{\Phi_i} \cdot Z_{\Phi_i}. \tag{13}$$

Then we can apply the final embedding to specific tasks and learn the attention value via the back propagation algorithm. For example, in node classification, we try to minimize the *Cross Entropy* between the ground-truth and the predictions:

$$L_{att} = -\sum y_i \log(wz_i), \tag{14}$$

where w is the parameter of the classifier, y_i is the label of z_i. With the guide of labeled data, we can optimize the proposed model and learn the weights of meta paths. Here we only need a few labeled data to fine-tune the pre-trained AutoEncoders.

Table 1. Statistics of the datasets

Dataset	Relations (A-B)	Number of A	Number of B	Number of A-B	Avg. degree of A	Avg. degree of B
DBLP	Paper-Author	14328	4057	19645	1.3	4.8
	Paper-Conf	14328	20	14328	1.0	716.4
	Paper-Term	14327	8789	88420	6.2	10.1
Yelp	Business-City	4352	253	4352	1.0	17.2
	Business-Tip	4352	41359	52262	12.0	1.3
	Tip-User	41359	25608	42262	1.0	1.7
	User-Review	125684	176860	176860	1.4	1.0
	Review-Business	176860	4352	176860	1.0	40.6

5 Experiments

5.1 Datesets

To verify the effectiveness of the proposed AMPE, we conduct some experiments on two real-world datasets. The detailed descriptions of these datasets are shown in Table 1, and their schemas are shown in Fig. 1.

- **DBLP**[1]. We extracted a subset of DBLP which contains papers (P), authors (A), conferences (C), terms (T). We obtain the ground truth from the dataset $dblp-4area$ [4], which labels each author according to their research area. Here we employ the meta path set {APA, APCPA, APTPA} to extract homogeneous networks.
- **Yelp**[2]. We extracted businesses located in North Carolina (NC), Wisconsin (WI), Pennsylvania (PA) and Edinburgh (EDH). Then we constructed a HIN that comprises businesses (B), users (U), cities (C), reviews (R) and tips (T). We employ the meta path set {BCB, BRURB, BTUTB} to extract homogeneous networks. Here we use the state information provided in the dataset as the ground truth.

5.2 Baselines

We compare with the following network embedding methods:

- DeepWalk [10]: A random walk based network embedding method for homogeneous network. Here we run DeepWalk on whole HIN and ignore the heterogeneity of nodes.
- AE [15]: A deep AutoEncoder (AE) that can embed networks via a series of non-linear mappings. Here we only report the best result of single meta path.

[1] https://dblp.uni-trier.de.
[2] https://www.yelp.com/dataset/download.

- AE$_{concat}$: A variant of the AutoEncoder model. We first apply AutoEncoders to learn node representations for each meta path, and concatenate them as the final representation.
- Metapath2vec [6]: A heterogeneous network embedding method which can embed the semantic information extract from a single meta path. Here we only report the best result of single meta path.
- ESim [8]: A heterogeneous network embedding method which can capture semantics information from multiple meta paths. Since it is difficult to search the weights of a set of meta paths, we assign the weights learned from AMPE to ESim.
- AMPE$_{avg}$: A variant of the proposed AMPE. We treat all meta paths equally and average the learned embeddings.
- AMPE: Our proposed approach for heterogeneous network embedding, which can fuse multiple meta paths according to their importances.

5.3 Parameter Settings

For DBLP, the architecture of AutoEncoder is 4057-1000-100-1000-4057 and β is set to 30. We use RMSprop to optimize AMPE and the learning rate is 0.001. Here we select 400 labeled data to fine-tune the AutoEncoder and learn the weights of meta paths. For Yelp, the architecture of AutoEncoder is 4352-100-4352. Here we utilize 200 balanced labeled data to fine-tune the model. Since such an AutoEncoder already work well, we don't need a deeper model.

For DeepWalk and metapath2vec, we set window size as 5, walk length as 20, walks per node as 40, num of negative samples as 5. For a fair comparison, we set embedding dimension as 100 for all above algorithms.

Table 2. Quantitative results on the node classification task

Datasets	Algorithms	30%		50%		70%	
		Macro-F1	Micro-F1	Macro-F1	Micor-F1	Macro-F1	Micro-F1
DBLP	DeepWalk	0.7456	0.7488	0.7785	0.7785	0.7930	0.7947
	AE	0.8928	0.8931	0.8978	0.8979	0.8894	0.8998
	AE$_{concat}$	0.8889	0.9055	0.9068	0.9068	0.9086	0.9086
	metapath2vec	0.8894	0.8889	0.8910	0.8905	0.8991	0.8998
	ESim	0.9135	0.9144	0.9125	0.9134	0.9172	0.9177
	AMPE$_{avg}$	0.9042	0.9041	0.9087	0.9085	0.9136	0.9138
	AMPE	**0.9237**	**0.9239**	**0.9239**	**0.9239**	**0.9248**	**0.9248**
Yelp	DeepWalk	0.8970	0.8970	0.9189	0.9233	0.9333	0.9378
	AE	0.9218	0.9272	0.9500	0.9522	0.9606	0.9642
	AE$_{concat}$	0.9491	0.9639	0.9808	0.9844	0.9846	0.9879
	metapath2vec	0.9728	0.9690	0.9760	0.9718	0.9742	0.9688
	Esim	0.9818	0.9793	0.9882	0.9890	0.9890	0.9890
	AMPE$_{avg}$	0.9510	0.9673	0.9682	0.9771	0.9746	0.9824
	AMPE	**0.9855**	**0.9866**	**0.9926**	**0.9927**	**0.9968**	**0.9965**

5.4 Classification

We start by conducting a task of multi-label classification to evaluate the effectiveness of the proposed model. First, we randomly divide the dataset into training set and test set. Then, we utilize KNN classifier with $k = 5$ to predict the labels of the test samples. We repeat the process for 10 times and report the average Macro-F1 and Micro-F1. The results are presented in Table 2.

The results show that, by distinguishing the importances of meta paths, AMPE achieves the best performance. Compare to HIN embedding methods, homogeneous network embedding methods including DeepWalk and AutoEncoder fail to perform well. For HIN embedding, through integrating multiple meta paths to perform HIN embedding, ESim and AMPE perform much better than other methods. Taking one step further, with weight learning for meta paths, AMPE performs much better than AMPE$_{avg}$. Overall, by utilizing deep model to embed and fusing semantics information extracted from HIN, AMPE achieves the best results in node classification task.

Table 3. Quantitative results on the node clustering task

Algorithms	DBLP	Yelp
DeepWalk	0.6656	0.6460
AE	0.6647	0.6631
AE$_{concat}$	0.6874	0.6386
metapath2vec	0.7195	0.8563
ESim	0.6205	0.6292
AMPE$_{avg}$	0.5225	0.6578
AMPE	**0.7474**	**0.9681**

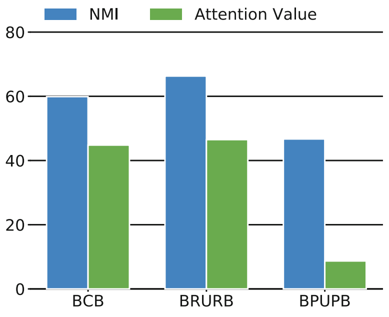
5.5 Clustering

We also conduct clustering task to evaluate the embeddings learned from above algorithms. Here we introduce the K-Means to perform node clustering and use NMI to evaluate the performances. Since the performance of K-Means is affected by initial centroids, we repeat the process for 10 times and report the average results. The results are shown in Table 3.

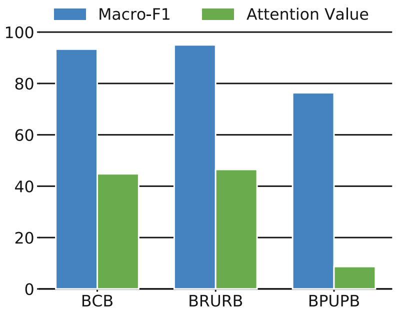
It's obviously that AMPE performs better than all baselines. Similar to the node classification, homogeneous network embedding methods fail to perform well. It's interesting that AMPE$_{avg}$ has the worst performance in DBLP. We will explain this phenomenon by analysing the attention value in next section. Besides, by concatenating the embeddings learned from different meta paths, the AE$_{concat}$ also performs well. Generally speaking, AMPE can give a comprehensive description of HIN.

5.6 Analysis of Attention Mechanism

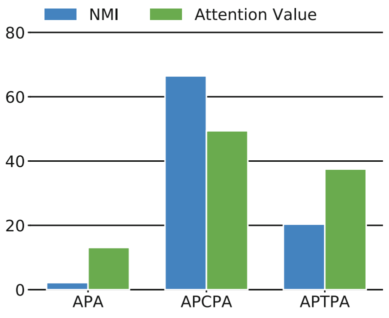
An interesting characteristic of AMPE is that it can learn the importances of meta paths via attention mechanism. In Fig. 3, we record the performances based on single meta path and corresponding attention value. For DBLP, AMPE gives APCPA the highest weight, which means AMPE considers the APCPA to be the most important meta path in identifying the author's research area. This is reasonable because we labeled the authors according to the conferences they submit. Meanwhile, APA cannot identify the author's research area. If we treat these meta paths equally (e.g., AMPE$_{avg}$), the performance will drop significantly. For Yelp, the results show that AMPE gives the largest weight to BRURB. One possible explanation is that users usually visit and review the businesses which is located near to their home for convenience. Obviously, there is a positive correlation between performance of single meta path and its attention value. It proves that the proposed AMPE can reveal the difference among these meta paths and weights them properly.



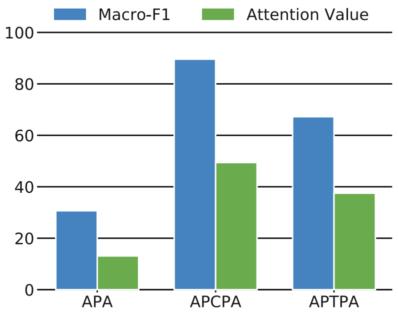
(a) NMI values on Yelp.



(b) Macro-F1 on Yelp.



(c) NMI values on DBLP.



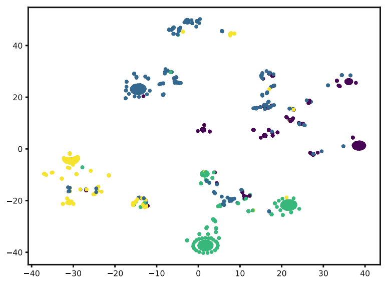
(d) Macro-F1 on DBLP.

Fig. 3. Performance of single meta path and corresponding attention value.

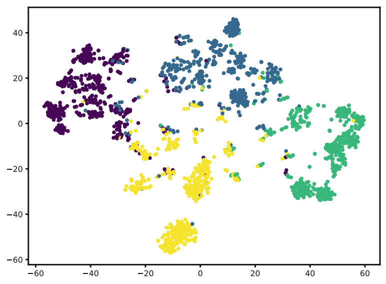
5.7 Visualization

For a more intuitive comparison, we visualize the learned embeddings on a 2-dimensional space. Here we utilize t-SNE to visualize the author embedding in DBLP. The results are shown in Fig. 4.

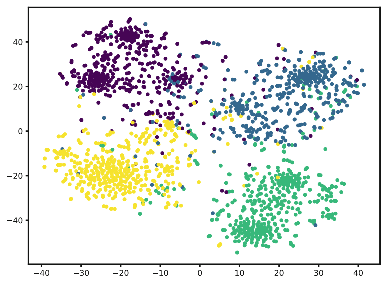
It's obvious that single AutoEncoder barely separates the authors from different groups (represented by the same color), but the distribution of blue points are scattered. Deepwalk can basically separate the authors without sharp boundaries. Metapath2vec performs much better than above methods, but the boundary is still blurry. Finally, AMPE achieves the best performance among these methods, since it can separate the authors in different research area clearly with obvious borders among these clusters.



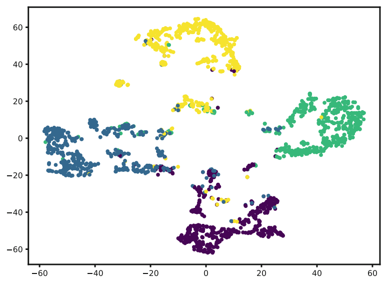
(a) AutoEncoder.



(b) DeepWalk.



(c) Metapath2vec.



(d) AMPE.

Fig. 4. Visualization results on DBLP. (Color figure online)

5.8 Parameters Experiments

Here we investigate the sensitivity of different parameters in our model, including embedding dimension and non-zero element penalty β. We first compare the performances of AMPE on DBLP with various numbers of embedding dimensions.

The results are shown in Fig. 5. We can see that with the growth of embedding dimension, the performance raises first and then remains stable. The reason is that AMPE needs a suitable dimension to encode these semantics information and larger dimension may introduce some redundancies. Then we show how the β affects the performances. Large β means that the model will pay more attention to non-zero elements. From the Fig. 5, We can see that AMPE achieves the best performance with a balanced β.

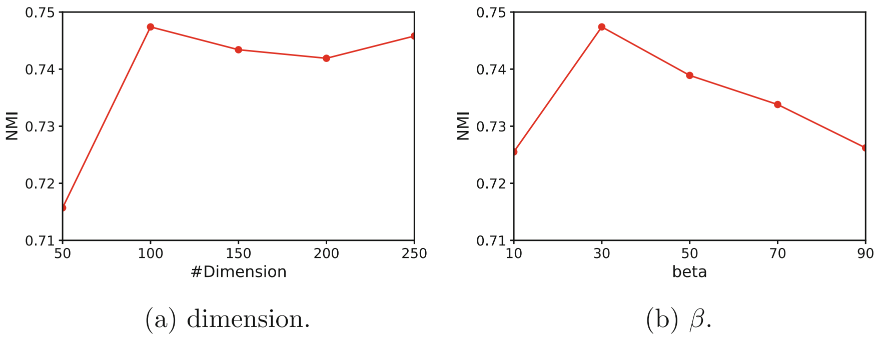


Fig. 5. Parameter study on the number of embedding dimensions and the value of non-zeros element penalty β.

6 Conclusion

In this paper, we study the problem of heterogeneous information network embedding, which aims to embed heterogeneous information network into a low-dimensional space. And we propose a novel heterogeneous information network embedding model, named AMPE, which can capture rich semantics information in HIN and fuses them for specific tasks. By extending the AutoEncoder to the heterogeneous scenario, AMPE can embed the semantics information extracted by meta paths simultaneously. In addition, the proposed AMPE can combine these information via attention mechanism. Experiment results including node classification and node clustering demonstrate the effectiveness of AMPE.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No. 61772082, 61375058), the National Key Research and Development Pro- gram of China (2017YFB0803304), and the Beijing Municipal Natural Science Foundation (4182043).

References

1. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. arXiv preprint arXiv:1711.08752 (2017)
2. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: a survey. arXiv preprint arXiv:1705.02801 (2017)
3. Zhang, D., Yin, J., Zhu, X., Zhang, C.: Network representation learning: a survey. arXiv preprint arXiv:1801.05852 (2017)
4. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: PathSim: meta path-based top-k similarity search in heterogeneous information networks. Proc. VLDB Endow. **4**(11), 992–1003 (2011)
5. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD Explor. Newsl. **14**(2), 20–28 (2013)
6. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 135–144. ACM (2017)
7. Fu, T.Y., Lee, W.C., Lei, Z.: Hin2vec: explore meta-paths in heterogeneous information networks for representation learning. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1797–1806. ACM (2017)
8. Shang, J., Qu, M., Liu, J., Kaplan, L.M., Han, J., Peng, J.: Meta-path guided embedding for similarity search in large-scale heterogeneous information networks. CoRR abs/1610.09769 (2016)
9. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
10. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710. ACM (2014)
11. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
12. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, pp. 1067–1077 (2015)
13. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225–1234. ACM (2016)
14. Sun, Y., Yu, Y., Han, J.: Ranking-based clustering of heterogeneous information networks with star network schema. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 797–806. ACM (2009)
15. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: Advances in Neural Information Processing Systems, pp. 153–160 (2007)
16. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)