Dynamic Heterogeneous Graph Embedding Using Hierarchical Attentions

Luwei Yang(✉), Zhibo Xiao, Wen Jiang, Yi Wei, Yi Hu, and Hao Wang

Alibaba Group, Hangzhou, China
{luwei.ylw,xiaozhibo.xzb,wen.jiangw}@alibaba-inc.com

Abstract. Graph embedding has attracted many research interests. Existing works mainly focus on static homogeneous/heterogeneous networks or dynamic homogeneous networks. However, dynamic heterogeneous networks are more ubiquitous in reality, e.g. social network, e-commerce network, citation network, etc. There is still a lack of research on dynamic heterogeneous graph embedding. In this paper, we propose a novel dynamic heterogeneous graph embedding method using hierarchical attentions (DyHAN) that learns node embeddings leveraging both structural heterogeneity and temporal evolution. We evaluate our method on three real-world datasets. The results show that DyHAN outperforms various state-of-the-art baselines in terms of link prediction task.

Keywords: Graph embedding · Heterogeneous network · Dynamic graph embedding

1 Introduction

Graph (Network) embedding has attracted tremendous research interests. It learns the projection of nodes in a network into a low-dimensional space by encoding network structures or/and node properties. This technique has been successfully applied to various domains, such as recommendation [11,18], node classification [8], link prediction [1] and biology [7].

In real-world, graphs often not only evolve over time but also contain multiple types of nodes and edges. For instance, e-commerce network has two types of nodes, user and item, and multiple types of edges, click, buy, add-to-preference and add-to-cart. The nodes and edges may change over time. In social network, users may develop their multiple-type connections (follow, reply, retweet, etc) with others over time. The dynamics of a network and the structural heterogeneity provide abundant information for encoding nodes.

Recent research mainly focuses on static graph embedding which has a fixed set of nodes and edges. DeepWalk [9] and node2vec [6] leverage a random walk/biased random walk and skip-gram model. LINE [12] preserves both first-order and second-order proximities. GCN [8] uses convolutional operations on node's neighborhood. GraphSAGE [7] or PinSAGE [18] proposes an inductive method

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 425–432, 2020.
https://doi.org/10.1007/978-3-030-45442-5_53

to aggregate structural information with node features. Further works consider heterogeneity. metapath2vec [2] takes meta-path into account when generating random walks. GATNE [1] aggregates node embedding by separating network into different views according to edge types. HAN [16] uses two-level attentions to learn the importance of neighbor nodes and meta-paths.

Dynamic graph embedding is an emerging area [17]. DynamicTriad [19] uses triadic closure to improve node embeddings. DySAT [10] extends the original GAT [15] to temporal graph snapshots. MetaDynaMix [4] proposes a metapath-based technique for dynamic heterogeneous information network embedding. More works may refer to [3,5,13].

Nonetheless, there is still a lack of research taking into account both temporal evolution and structural heterogeneity. Inspired by works on [16] and [10], we propose a novel dynamic heterogeneous graph embedding approach using hierarchical attention layers (DyHAN), which is able to capture the importance in different level aggregations. To be specific, for an arbitrary node, node-level attention intends to learn the importance of its neighbor for a specific edge type. Edge-level attention aims to learn the importance of every edge-type for this node. Temporal-level attention is able to fuse the final embedding by figuring out the importance of each time step graph snapshot. We evaluate our method on three real-world dynamic heterogeneous network datasets, EComm, Twitter and Aliaba.com. The results show that DyHAN outperforms several state-of-the-art baselines in link prediction task.

2 Problem Definition

In this section, we provide necessary information throughout this paper. We consider a dynamic heterogeneous network is defined as a series of snapshots, $\{G^1, G^2, ..., G^T\}$. A snapshot at time t is defined as $G^t = (\mathcal{V}^t, \mathcal{E}^t, \mathcal{W}^t)$, where $\mathcal{V}^t$ is the node set with node type $o \in \mathcal{O}$. $\mathcal{E}^t$ is the edge set with edge type $r \in \mathcal{R}$. $\mathcal{O}$ and $\mathcal{R}$ are node type set and edge type set respectively, and $|\mathcal{O}| + |\mathcal{R}| > 2$. We assume for each time snapshot the nodes and links both can be changed.

Dynamic heterogeneous graph embedding aims to learn a mapping function $f : \mathcal{V} \rightarrow \mathbb{R}^d$, such that it preserves the structural similarity among nodes and their temporal tendencies in developing link relationships.

3 Proposed Method

In this section, we introduce our proposed approach DyHAN employing hierarchical attentions on dynamic heterogeneous graph embedding which combines the basic ideas proposed in [10,16]. It has three main components, node-level attention, edge-level attention and temporal-level attention. All of these three components aggregate different layer of information using different attention layers. The overall architecture of DyHAN is represented by Fig. 1.

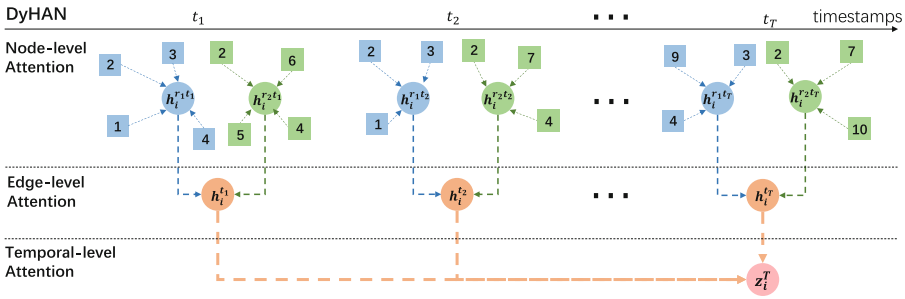

Fig. 1. Architecture of DyHAN.

Node-Level Attention. For each time step snapshot, we separate it into different subgraphs according to edge types. A self-attention is employed to aggregate node embedding for each subgraph. The importance of node pair (i, j) for edge type r and time step t can be expressed by,

$$\alpha_{ij}^{rt} = \frac{\exp(\sigma(\mathbf{a}_r^\top [\mathbf{W}_{nl}^r \mathbf{x}_i || \mathbf{W}_{nl}^r \mathbf{x}_j]))}{\sum_{k \in N_i^{rt}} \exp(\sigma(\mathbf{a}_r^\top [\mathbf{W}_{nl}^r \mathbf{x}_i || \mathbf{W}_{nl}^r \mathbf{x}_k]))}, \tag{1}$$

where σ is an activation function, $\mathbf{x}_i$ is the input representation of node i, $\mathbf{W}_{nl}^r$ is a linear transformation matrix, $||$ denotes the concatenation. N_i^{rt} denotes the sampled neighbor nodes for node i for edge type r and time step t. Different from [15] which uses all immediate neighbors, instead, for the sake of induction, we follow the framework described in [7] to use the sampled neighbors. $\mathbf{a}_r$ is a weight vector that parameterizes the attention function for edge type r. Then the embedding of node i for edge type r and time step t is obtained as,

$$\mathbf{h}_i^{rt} = \sigma\left(\sum_{j \in N_i^{rt}} \alpha_{ij}^{rt} \cdot \mathbf{W}_{nl}^r \mathbf{x}_j\right). \tag{2}$$

Note that the parameters are shared among different time step snapshots.

Edge-Level Attention. We assume the edge-specific node embedding expresses one semantic type of information in a heterogeneous graph. To aggregate these information more efficiently and robustly, we employ an attention layer to learn the importance of different edge types automatically. The importance of each edge type is calculated by an one-layer MLP.

$$\beta_i^{rt} = \frac{\exp(\mathbf{q}^\top \cdot \sigma(\mathbf{W}_{el} \mathbf{h}_i^{rt} + \mathbf{b}_{el}))}{\sum_{l=1}^{R} \exp(\mathbf{q}^\top \cdot \sigma(\mathbf{W}_{el} \mathbf{h}_i^{lt} + \mathbf{b}_{el}))} \tag{3}$$

where σ is an activation function, $\mathbf{q}^\top$ is the edge-level attention vector. $\mathbf{W}_{el}$ and $\mathbf{b}_{el}$ are the one-layer MLP's parameters. All parameters are shared across

different time steps and different edge types. Then the fused embedding of node i is,

$$\mathbf{h}_i^t = \sum_{r=1}^{R} \beta_i^{rt} \cdot \mathbf{h}_i^{rt}. \tag{4}$$

Temporal-Level Attention. Once obtained the node embeddings for each time step snapshot, the next is to aggregate these node embeddings across a series of time snapshots. To compute the final node embedding, we use $\mathbf{h}_i^T$ to attend over all its historically-temporal representations, $\{\mathbf{h}_i^1, \mathbf{h}_i^2, ..., \mathbf{h}_i^{T-1}\}$. The Scaled Dot-Product Attention [14] is used by assuming that it is able to capture temporal evolution characteristics. We pack the representation of node i across time as $\mathbf{H}_i \in \mathbb{R}^{T \times D}$. Then, $\mathbf{H}_i$ is transformed into queries $\mathbf{Q} = \mathbf{H}_i \mathbf{W}_q$, keys $\mathbf{K} = \mathbf{H}_i \mathbf{W}_k$ and values $\mathbf{V} = \mathbf{H}_i \mathbf{W}_v$, where $\mathbf{W}_q \in \mathbb{R}^{D \times D'}$, $\mathbf{W}_k \in \mathbb{R}^{D \times D'}$ and $\mathbf{W}_v \in \mathbb{R}^{D \times D'}$. The temporal attention is defined as,

$$\mathbf{Z}_i = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D'}} + \mathbf{M}) \cdot \mathbf{V}, \tag{5}$$

where $\mathbf{M} \in \mathbb{R}^{T \times T}$ is a mask matrix so that $\mathbf{h}_i$ only attends over time steps $\leq t$.

$$M_{ij} = \begin{cases} 0 & \text{if } i \leq j, \\ -\infty, & \text{otherwise} \end{cases} \tag{6}$$

We will use the $\mathbf{z}_i^T$ as the final node embedding. Note that multi-head attention could be applied to node-level and temporal-level attentions.

Optimization. In order to train the model capturing both structural and temporal information, we encourage nearby nodes at the last time step to have similar representations. A cross entropy loss is employed,

$$L(\mathbf{z}_u^T) = -\log(\sigma(< \mathbf{z}_u^T, \mathbf{z}_v^T >)) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log(\sigma(< -\mathbf{z}_u^T, \mathbf{z}_{v_n}^T >)) \tag{7}$$

where σ is the sigmoid function and $<,>$ denotes the inner product. v is the node that co-occurs near u on fixed-length random walk in the last time step. P_n is a negative sampling distribution, here we use the node's degree in the last time step. Q defines the number of negative samples.

4 Experiments

Datasets. We use three real-world datasets for evaluation. The statistics of them are summarized in Table 1.

EComm[1] dataset is sampled from the dataset of CIKM 2019 EComm AI contest from a category. There are two types of nodes, user and item. It has four types of edges including click, collect, add-to-cart and buy.

[1] https://tianchi.aliyun.com/competition/entrance/231719/introduction.

Twitter[2] dataset is sampled from the user behavior logs in Twitter about the discovery of elusive Higgs boson between 1st and 7th July 2012. There are three types of edges: retweet, reply and mention. Note that there is only one type of node.

Alibaba.com dataset is sampled from the user behavior logs in the alibaba.com e-commerce platform. A network from customer electronics category between 11th July and 21st July 2019 is sampled. It consists of interactions between users and items. There are three types of interactions, click, enquiry and contact.

Table 1. Statistics of datasets.

Dataset	# nodes	# edges	# node types	# edge types	# time steps
EComm	37724	91033	2	4	11
Twitter	100000	63410	1	3	7
Alibaba.com	16620	93956	2	3	11

Experimental Setup. We learn node embeddings based on graph snapshots $\{G^1, G^2, ..., G^t\}$, then a link prediction experiment is conducted on the last graph snapshot G^{t+1}.

A link prediction task aims to predict whether there is an existing link between any two nodes. We follow the evaluation framework for link prediction as stated in [10,19]. We create a Logistic Regression classifier for dynamic link predictions. We sample 20% of edges from the last time step snapshot as the held-out validation set for hyper-parameter tuning. The rest of edges of the last time step snapshot are used for link prediction task. In specific, we choose randomly 25% of links and the remaining 75% of links as training and test set respectively. An equal number of randomly sampled pairs of nodes without link as negative examples for each training and test set respectively. We use the inner product of the node embeddings of the node-pair as the representation feature of the link. Then Area Under the ROC Curve (AUC) [9] score and accuracy are used to report the performance.

Baselines. Considering availability of code and the effort of reimplementation, we compare our proposed DyHAN with following state-of-the-art static/dynamic and homogeneous/heterogeneous graph embedding algorithms. *DeepWalk* [9], we use the implementation provided by [7]. *Metapath2Vec* [2], the original implementation provided by the authors are dedicated to specific dataset. As a result, it is not convenient to directly generalize to other datasets. We reimplemented it in python. *GAT* [15], the original implementation provided by the authors is designed for node classification. We reimplemented it in the GraphSAGE framework. Note that the nodes to be attended over are sampled from immediate neighbors. *GraphSAGE* [7], we use the implementation provided by the

[2] http://snap.stanford.edu/data/higgs-twitter.html.

authors and use the default settings. Four variants with different node aggregation techniques are tested, namely, mean, mean-pooling, max-pooling and LSTM. *DynamicTriads* [19] and *DySAT* [10], we use the implementation provided by the authors. A method named *DyGAT* which ignores the structural heterogeneity was also implemented for comparison of incorporating heterogeneity. For random-walk based methods, we set the number of walks for each node as 50 and the length of each walk is set to 5. All training epoch is set to 1. All node embedding dimension is set to 32.

Results. The experimental results are shown by Table 2. DyHAN achieves the highest AUC score and accuracy among competitors. To be more specific, DyHAN obtains gains of 2.8%–4.9% on AUC and gains of 0.7%–7.8% on accuracy comparing the best baseline (exluding DyGAT). The gains of DyGAT over GAT show the efficacy of incorporating temporal information. Furthermore, the gains of DyHAN over DyGAT shows the efficacy of considering heterogeneity.

Table 2. Experimental results on three real-world datasets.

Method	EComm		Twitter		Alibaba.com	
	ROC-AUC	Accuracy	ROC-AUC	Accuracy	ROC-AUC	Accuracy
DeepWalk	0.573	0.554	0.571	0.661	0.558	0.538
Metapath2Vec	0.613	0.574	0.571[a]	0.661[a]	0.577	0.570
GraphSAGE-mean	0.640	0.602	0.557	0.640	0.549	0.533
GraphSAGE-meanpool	0.584	0.554	0.574	0.661	0.571	0.547
GraphSAGE-maxpool	0.638	0.606	0.559	0.622	0.568	0.551
GraphSAGE-LSTM	0.579	0.551	0.564	0.635	0.563	0.542
GAT	0.656	0.601	0.580	0.634	0.557	0.533
DynamicTriad	0.595	0.567	0.641	0.661	0.571	0.524
DySAT	0.504	0.496	0.652	0.661	0.523	0.527
DyGAT	0.680	0.638	0.645	0.633	0.569	0.539
DyHAN	**0.688**	**0.653**	**0.659**	**0.672**	**0.601**	**0.574**

[a]Note that Metapath2Vec is same as DeepWalk when the number of node type is one.

5 Conclusions

In this paper, we have proposed a novel hierarchical attention neural networks named DyHAN to learn node embeddings in dynamic heterogeneous graphs. DyHAN is able to effectively capture both structural heterogeneity and temporal evolution. Experimental results on three real-world datasets show that DyHAN outperforms several state-of-the-art techniques. One interesting future direction is exploring more temporal aggregation techniques.

References

1. Cen, Y., Zou, X., Zhang, J., Yang, H., Zhou, J., Tang, J.: Representation learning for attributed multiplex heterogeneous network. In: Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2019)
2. 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 (2017)
3. Du, L., Wang, Y., Song, G., Lu, Z., Wang, J.: Dynamic network embedding: an extended approach for skip-gram based network embedding. In: Proceedings of the Twenty-Seventh International Joint Conference on Aritificial Intelligence (2018)
4. Milani Fard, A., Bagheri, E., Wang, K.: Relationship prediction in dynamic heterogeneous information networks. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) ECIR 2019. LNCS, vol. 11437, pp. 19–34. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15712-8_2
5. Goyal, P., Kamra, N., He, X., Liu, Y.: DynGEM: Deep embedding method for dynamic graphs. arXiv preprint arXiv:1805.11273 (2018)
6. 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 (2016)
7. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, pp. 1024–1034 (2017)
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: Proceedings of the International Conference on Learning Representations (2017)
9. 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 (2014)
10. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: Dynamic graph representation learning via self-attention networks. In: Proceedings of Workshop on Representation Learning on Graphs and Manifolds in the Seventh International Conference on Learning Representation (2019)
11. Shi, C., Hu, B., Zhao, W.X., Yu, P.S.: Heterogeneous information network embedding for recommendation. IEEE Trans. Knowl. Data Eng. **31**(2), 357–370 (2018)
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 (2015)
13. Trivedi, R.S., Farajtabar, M., Biswal, P., Zha, H.: Learning dynamic graph representations. In: Proceedings of Workshop on Modeling and Decision-Making in the Spatiotemporal Domain in the 32nd Conference on Neural Information Processing Systems (2018)
14. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
15. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: Proceedings of the 6th International Conference on Learning Representations (2018)
16. Wang, X., et al.: Heterogeneous graph attention network. In: Proceedings of the Web Conference (2019)

17. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596 (2019)
18. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J.: Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974–983 (2018)
19. Zhou, L., Yang, Y., Ren, X., Wu, F., Zhuang, Y.: Dynamic network embedding by modeling triadic closure process. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (2018)