M-HIN: Complex Embeddings for Heterogeneous Information Networks via Metagraphs

Yang Fang
National University of Defense Technology, China
fangyang12@nudt.edu.cn

Xiang Zhao
National University of Defense Technology, China
xiangzhao@nudt.edu.cn

Peixin Huang
National University of Defense Technology, China
huangpeixin15@nudt.edu.cn

Weidong Xiao
National University of Defense Technology, China
wdxiao@nudt.edu.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

To represent a complex network, paths are often employed for capturing relationships among node: random walks for (homogeneous) networks and metapaths for heterogeneous information networks (HINs). However, there is structural (and possibly semantic) information loss when using paths to represent the subgraph between two nodes, since a path is a linear structure and a subgraph often is not. Can we find a better alternative for network embeddings? We offer a novel mechanism to capture the features of HIN nodes via metagraphs, which retains more structural and semantic information than path-oriented models. Inspired by developments in knowledge graph embedding, we propose to construct HIN triplets using nodes and metagraphs between them. Metagraphs are generated by harnessing the GRAMI algorithm, which enumerates frequent subgraph patterns in a HIN. Subsequently, the Hadamard function is applied to encode relationships between nodes and metagraphs, and the probability whether a HIN triplet can be evaluated. Further, to better distinguish between symmetric and asymmetric cases of metagraphs, we introduce a complex embedding scheme that is able to precisely express fine-grained features of HIN nodes. We evaluate the proposed model, M-HIN, on real-life datasets and demonstrate that it significantly and consistently outperforms state-of-the-art models.

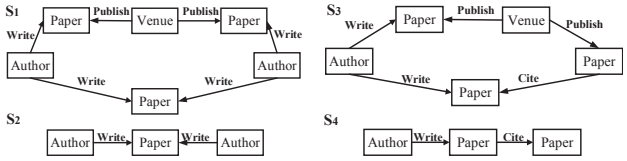
KEYWORDS

Heterogeneous information network; Representation learning; Network embedding; Metagraph

ACM Reference Format:
Yang Fang, Xiang Zhao, Peixin Huang, Weidong Xiao, and Maarten de Rijke. 2019. M-HIN: Complex Embeddings for Heterogeneous Information Networks via Metagraphs. In *Proceedings of The 42nd ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19).* ACM, New York, NY, USA, 4 pages. https://doi.org/xx.xxx/xxx_x

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/xx.xxx/xxx_x



(a) Symmetric metagraphs (b) Asymmetric metagraphs

Figure 1: Examples of metagraphs.

1 INTRODUCTION

A *heterogeneous* information network (HIN) is a network with nodes and edges of multiple types. HINs serve as a richer tool to model real-life problems than homogeneous networks, as most networks come with multiple types of nodes and edges.

To be able to work with HINs in a machine learning context, *network representation learning*, also known as network embedding learning, which embeds a network into a low-dimensional space, has been investigated extensively. Classic network embedding models [6, 8, 9] that rely on random walks suffer from low capability of representing HINs, and thus, HIN-specific embedding models have been proposed. Existing HIN embedding models are mainly built on the basis of *metapath*, which is a sequence of node types with edge types. For example, in Figure 1(a), S_2 is a metapath and it represents a co-author relationship. Different mechanisms have been devised to model relationships between HIN nodes, e.g., proximity distance [7], heterogeneous SkipGram [2], and the Hadamard function [5], etc. Using either random walks or metapaths is, however, insufficient to describe the neighborhood structure of a node in a complete manner, since there is always an apparent amount of information loss when extracting and employing paths to represent neighborhood graphs. For example, in Figure 1(a), S_2 is only a linear structure that merely captures the co-author relationship, and hence, ignores other relationships between two author nodes, e.g., shared-venue. In other words, existing models are not expressive enough to retain as many features of a HIN as possible. Therefore, a pressing research question is whether there is a better alternative for representing HINs.

To answer the question listed above, we put forward a novel HIN embedding model, namely, M-HIN, which provides a new perspective to capture the features of a HIN via *metagraphs*. A

metagraph is a subgraph of node types with edge types in between, which is able to frame the connecting subgraph of two node types, and hence, completely retains the neighborhood structure of nodes; toy examples are provided in Figure 1. We can observe that in Figure 1(a), unlike metapaths, a metagraph can *jointly* model multiple relationships; for example, S_1 describes co-author and shared-venue relationships between two authors, thus having more semantic information than S_2. Actually, a metapath is a special kind of metagraph. Inspired by recent advance in representation learning for knowledge graphs, M-HIN explicitly constructs *HIN triplets* in the form of (u, s, v), where u and v are HIN nodes, and s denotes the metagraph connecting u to v. Compared with existing models relying on paths, metagraphs preserve more abundant structure and capture more accurate features of nodes [4]. Subsequently, the Hadamard function and `sigmoid` function are leveraged to evaluate the probability of a HIN triplet being appropriate.

When training the model, we observe that the structure of metagraphs could be either symmetric or asymmetric, as illustrated in Figures 1(a) and 1(b), respectively. The Hadamard function is not able to describe such complicated relationships between nodes and metagraphs. To better represent HIN nodes, we choose to further incorporate a complex space oriented embedding scheme [10] to handle symmetric and asymmetric relationships between nodes. In complex space, the entries in the embedding vectors of nodes are complex-valued; that is, we separate the vectors of nodes into real and imaginary parts, and transfer the Hadamard function into complex space. Consequently, when s is purely imaginary, the entire function is correspondingly antisymmetric; when s is real, it is symmetric; and similar properties hold for embeddings of metagraphs.

To evaluate M-HIN, we perform experiments on two benchmark tasks, node classification and link prediction. M-HIN consistently outperforms state-of-the-art models

2 RELATED WORK

Our research falls into the area of network embeddings, and we list recent developments in the area. DeepWalk [8] leverages random walks and applies the SkipGram model to learn network embedding. node2vec [6] is an extension of DeepWalk, as it adopts a biased random walk strategy that can better explore the network structure. LINE [9] harnesses first-order and second-order proximities simultaneously to encode local and neighborhood structure information.

While the aforementioned approaches are designed for homogeneous networks, there is also dedicated research specifically exploiting the features of heterogeneous network. metapath2vec [2] proposes a heterogeneous SkipGram with its context window restricted to one specific type. HINE [7] proposes a metapath-based notion of proximity, and preserves proximity by minimizing the distance between nodes' joint probability as defined by sigmoid and empirical probabilities. HIN2Vec [5] devises the Hadamard multiplication of nodes and metapaths to capture features of a HIN, but symmetric and asymmetric structures inside a HIN are ignored. In a recent publication the term MetaGraph2Vec [11] has been coined. However, this publication utilizes heterogeneous SkipGrams and combines metapath to form so-called "metagraphs," which, in essence, is a path-oriented model.

3 PROPOSED MODEL

We first provide preliminaries, and then describe the proposed model, followed by the formulation of its training objective.

3.1 Preliminaries

First, we introduce the definitions of HIN and metagraph. A HIN is a directed graph $G = (V, E, T)$, where V denotes the set of nodes and E denotes the set of edges between nodes. Each node and edge is associated with a type mapping function, $\phi : V \rightarrow T_V$ and $\varphi : E \rightarrow T_E$, respectively. T_V and T_E denote the sets of node and edge types. A HIN is a network where $|T_V| > 1$ and/or $|T_E| > 1$. A *metagraph* is a subgraph of compatible node types and edge types, denoted as $S = (T_{VS}, T_{ES})$, where T_{VS} and T_{ES} denote the set of node types and edge types in a metagraph, respectively.

We use HIN triplets to demonstrate the relationship between nodes and metagraphs. Specifically, for a HIN triplet (u, s, v), where u is the first node generated in a metagraph, v the last and s is the metagraph connecting them, $\mathbf{u} \in \mathbb{C}^d$, $\mathbf{v} \in \mathbb{C}^d$ and $\mathbf{s} \in \mathbb{C}^d$ are the representation vectors of u, v and s, respectively, where d is the dimension of the representation vectors.

3.2 Metagraph HIN (M-HIN)

The probability of whether a HIN triplet holds is denoted as

$$P(s|u, v) = \sigma(\mathbf{X}_{uv}), \tag{1}$$

where $X \in \mathbb{R}^{n \times n}$ is a score matrix, n is the number of training nodes, and σ is an activation function and we choose the `sigmoid` function. Then we apply the Hadamard function to capture the relationship of u, v and s, which is represented as

$$\mathbf{X}_{uv} = \sum \mathbf{u} \odot \mathbf{s} \odot \mathbf{v}, \tag{2}$$

where $\sum$ is the element-wise summation function.

When training the model, we find that a metagraph may be symmetric or asymmetric, while the Hadamard function is not expressive enough to handle such complex relationships. If a metagraph is symmetric, replacing the first and last node will not change the semantics of the original metagraph, as shown in Figure 1(a). Correspondingly, replacing the head and tail node will change the semantics of an asymmetric metagraph, as shown in Figure 1(b).

To address this issue, we adapt the schema of complex knowledge embeddings in [10] to fit the task of network embeddings. For a HIN triplet (u, s, v), its complex embedding is denoted as $\mathbf{u} = \mathrm{Re}(\mathbf{u}) + i\mathrm{Im}(\mathbf{u})$, $\mathbf{v} = \mathrm{Re}(\mathbf{v}) + i\mathrm{Im}(\mathbf{v})$ and $\mathbf{s} = \mathrm{Re}(\mathbf{s}) + i\mathrm{Im}(\mathbf{s})$, where $\mathrm{Re}(\mathbf{x}) \in \mathbb{R}^d$ and $\mathrm{Im}(\mathbf{x}) \in \mathbb{R}^d$ represent the real and imaginary part of the vector $\mathbf{x} \in \mathbb{C}^d$, respectively. Thus, one element of the score matrix will be changed as

$$\mathbf{X}_{uv} = \sum \mathbf{u} \odot \mathbf{s} \odot \bar{\mathbf{v}}, \tag{3}$$

where $\bar{\mathbf{v}}$ is the complex conjugate form of $\mathbf{v}$. However, the sigmoid function in Eq. 1 cannot be applied in complex spaces, thus we only keep the real part of the objective function which is still able to handle the symmetric and asymmetric structures well, and we will illustrate the reason in detail later. So, one element in the score matrix will eventually be

$$\mathbf{X}_{uv} = \mathrm{Re}\left(\sum \mathbf{u} \odot \mathbf{s} \odot \bar{\mathbf{v}}\right). \tag{4}$$

Now that we have obtained the score matrix, the corresponding score function is defined as

$$\begin{aligned} f_{\mathbf{s}}(\mathbf{u},\mathbf{v}) &= \mathrm{Re}(\langle \mathbf{u},\mathbf{s},\bar{\mathbf{v}}\rangle) \\ &= \langle \mathrm{Re}(\mathbf{u}),\mathrm{Re}(\mathbf{s}),\mathrm{Re}(\mathbf{v})\rangle + \langle \mathrm{Im}(\mathbf{u}),\mathrm{Re}(\mathbf{s}),\mathrm{Im}(\mathbf{v})\rangle \\ &\quad + \langle \mathrm{Re}(\mathbf{u}),\mathrm{Im}(\mathbf{s}),\mathrm{Im}(\mathbf{v})\rangle - \langle \mathrm{Im}(\mathbf{u}),\mathrm{Im}(\mathbf{s}),\mathrm{Re}(\mathbf{v})\rangle, \end{aligned} \tag{5}$$

where $\langle\cdot\rangle$ is the standard element-wise multi-linear dot product. For example, $\langle a,b,c\rangle = \sum_k a_k b_k c_k$, where a, b, c are vectors and k represents the dimension of a vector.

Eq. 5 is able to handle asymmetric metagraphs thanks to the complex conjugate of one of the embeddings. In addition, if $\mathbf{s}$ is purely imaginary, i.e., its real part is zero, the score function is antisymmetric, and if real, the function is symmetric.[1] By separating the imaginary and real part of the metagraph embedding s, we obtain a decomposition of the metagraph matrix $\mathbf{X_s}$ as the sum of an antisymmetric matrix $\mathrm{Im}(\mathbf{U}\odot\mathrm{diag}(-\mathrm{Im}(\mathbf{s}))\odot\bar{\mathbf{V}})$ and a symmetric matrix $\mathrm{Re}(\mathbf{U}\odot\mathrm{diag}(\mathrm{Re}(\mathbf{s}))\odot\bar{\mathbf{V}})$. From this we can see that metagraph embeddings naturally act as weights on each latent dimension, i.e., $\mathrm{Im}(\mathbf{s})$ over the antisymmetric, imaginary part of $\langle\mathbf{u},\mathbf{v}\rangle$ and $\mathrm{Re}(\mathbf{s})$ over the symmetric, real part of $\langle\mathbf{u},\mathbf{v}\rangle$. Actually, we have that $\langle\mathbf{u},\mathbf{v}\rangle = \overline{\langle\mathbf{v},\mathbf{u}\rangle}$, meaning that $\mathrm{Im}(\langle\mathbf{u},\mathbf{v}\rangle)$ is antisymmetric, while $\mathrm{Re}(\langle\mathbf{u},\mathbf{v}\rangle)$ is symmetric. Therefore, the mechanism introduced above enables us to accurately and effectively represent both symmetric and asymmetric (including antisymmetric) metagraphs between pairs of nodes.

3.3 Training objective

We first generate metagraphs to build training datasets. Unlike MetaGraph2Vec [11] that simply combines several metapaths to construct their so-called "metagraphs," we directly adopt an existing state-of-the-art approach, viz., GRAMI [3], to find all subgraphs that appear frequently in a database according to a given frequency threshold. Then we use these mined subgraphs to form metagraphs.

To train M-HIN, we prepare the training datasets using a negative sampling strategy. For each positive HIN triplet (u,s,v), we generate the negative HIN triplets by randomly replacing u and v with other nodes, meanwhile restricting them to have the same type as the replaced one. We also filter out those replaced HIN triplets remaining positive after sampling. Notice that the number of candidates of s is much smaller than that of u or v, so the sampled negative data are generated only by replacing u and v.

After sampling, we have training data in the form of (u,s,v,Y_{uv}), where $Y_{uv}\in\{1,0\}$ is a binary value indicating whether the HIN triplet is positive or not. For a training instance (u,s,v,Y_{uv}), if $Y_{uv}=1$, the objective function $O_{(u,s,v)}$ aims to maximize $P(s|u,v)$; otherwise, $P(s|u,v)$ should be minimized. Thus, we have our objective function as follows

$$O_{(u,s,v)} = \begin{cases} P(s|u,v), & \text{if } Y_{uv}=1; \\ 1-P(s|u,v), & \text{if } Y_{uv}=0. \end{cases} \tag{6}$$

To simplify our computations, we define $\log O_{(u,s,v)}$ as follows

$$\log O_{(u,s,v)} = Y_{uv}\log P(s|u,v) + (1-Y_{uv})\log[1-P(s|u,v)], \tag{7}$$

where $P(s|u,v)$ is defined as

$$P(s|u,v) = \mathrm{sigmoid}(f_{\mathbf{s}}(\mathbf{u},\mathbf{v})). \tag{8}$$

[1] Mathematically, if f(x,y)=f(y,x), the function is symmetric, otherwise it is asymmetric, and in particular, if f(x,y)=-f(y,x), the function is antisymmetric.

Finally, we apply a stochastic gradient algorithm to maximize the above objective function.

4 EXPERIMENTS

4.1 Experimental setup

To evaluate the effectiveness of M-HIN to represent a HIN, we conduct experiments on four real-world datasets, extracted from DBLP,[2] YELP,[3] YAGO[4] and Freebase.[5] DBLP is a bibliographic dataset in computer science. Authors were separated into four areas: databases, machine learning, data mining and information retrieval. YELP is a social media dataset, which is about reviews on restaurants; the restaurants were split into three types—American, sushi bar and fast food. We also extracted a subset from YAGO containing knowledge and facts about movies; the movies were divided into five genres—horror, action, adventure, crime and sci-fi. The dataset extracted from Freebase is related to video games, consisting of games divided into three kinds—action, adventure, and strategy. Descriptive statistics are provided in Table 1. The datasets

Table 1: Dataset statistics.

Dataset	#nodes	#edges	# node types	# labels
DBLP	276,248	1,205,627	4	4
YELP	162,623	745,439	4	3
YAGO	25,643	40,173	5	4
Freebase	6,909	7,754	4	3

extracted from DBLP and YELP are much larger than the other two datasets, and we want to prove that M-HIN is scalable to both large and small datasets. The dataset sizes are in line with those having been reported in the existing literature.

We include three baselines, DeepWalk, LINE and node2vec, which were originally designed to represent homogeneous networks. For a fair comparison, we also include four models, metapath2vec, HINE, HIN2Vec and MetaGraph2Vec, devised for heterogeneous network embedding, which all leverage metapaths to capture features of a HIN, but in different ways. Moreover, we add another model M-HIN-path, which is a variant of M-HIN that merely employs metapaths (path patterns generated by GRAMI).

As to parameters for M-HIN and M-HIN-path, the dimensionality of node vectors was set to 128, the ratio of negative sampling was set to 5 (i.e., 5 negative samples for each positive sample), the learning rate in stochastic gradient descent was initialized to 0.025, and the training epoch was set to 10.

We report on statistical significance with a paired two-tailed t-test; we mark a significant improvement of M-HIN over HIN2Vec for $p<0.05$ with $^{\blacktriangle}$.

4.2 Node classification

In this section we report on the results for the multi-label node classification task. The labels for each dataset are introduced in Section 4.1. We calculate the micro-f1 (MIC-F1) and macro-f1 (MAC-F1) as evaluation metrics.

Table 2 provides the experimental results. The performance of all models is influenced by the scale of the datasets, and the outputs

[2] http://dblp.uni-trier.de
[3] https://www.yelp.com/dataset_challenge
[4] https://old.datahub.io/dataset/yago
[5] https://developers.google.com/freebase/

Table 2: Experimental results on the node classification task.

	DBLP		YELP		YAGO		Freebase	
Model	MIC-F1	MAC-F1	MIC-F1	MAC-F1	MIC-F1	MAC-F1	MIC-F1	MAC-F1
DeepWalk	0.197	0.194	0.167	0.151	0.330	0.277	0.542	0.484
LINE	0.186	0.184	0.278	0.280	0.368	0.321	0.520	0.449
node2vec	0.203	0.198	0.179	0.154	0.334	0.281	0.544	0.489
metapath2vec	0.211	0.213	0.265	0.271	0.371	0.332	0.517	0.436
HINE	0.235	0.233	0.281	0.285	0.401	0.366	0.519	0.438
HIN2Vec	0.247	0.242	0.295	0.307	0.432	0.396	0.561	0.502
MetaGraph2Vec	0.219	0.223	0.273	0.286	0.382	0.338	0.524	0.441
M-HIN-path	0.255	0.253	0.304	0.312	0.436	0.401	0.567	0.507
M-HIN	**0.262**▲	**0.259**▲	**0.309**	**0.321**▲	**0.443**▲	**0.408**▲	**0.574**▲	**0.513**▲

Table 3: Experimental results on the link prediction task.

	DBLP		YELP		YAGO		Freebase	
Model	MAP	R@100	MAP	R@100	MAP	R@100	MAP	R@100
DeepWalk	0.122	0.189	0.129	0.192	0.154	0.279	0.244	0.436
LINE	0.117	0.177	0.121	0.182	0.161	0.293	0.218	0.404
node2vec	0.128	0.194	0.133	0.201	0.166	0.288	0.237	0.427
metapath2vec	0.137	0.197	0.148	0.206	0.183	0.293	0.264	0.442
HINE	0.139	0.211	0.164	0.229	0.179	0.290	0.282	0.477
HIN2Vec	0.144	0.218	0.182	0.241	0.206	0.324	0.307	0.503
MetaGraph2Vec	0.141	0.203	0.153	0.217	0.187	0.297	0.271	0.457
M-HIN-path	0.151	0.224	0.185	0.246	0.213	0.332	0.315	0.527
M-HIN	**0.157**▲	**0.228**▲	**0.191**▲	**0.255**▲	**0.239**▲	**0.355**▲	**0.331**▲	**0.546**▲

become poorer when the datasets become larger. However, M-HIN outperforms all other models on every dataset. Specifically, HINE, metapath2vec and MetaGraph2Vec outperform the homogeneous network methods DeepWalk, LINE and node2vec but they perform slightly worse on Freebase, which indicates that metapaths are a better way to explore the features of a network than random walks. As mentioned in Section 2, MetaGraph2Vec is basically a path-oriented model, so it is only slightly better than metapath2vec with more metapaths combined. HIN2Vec performs consistently better than HINE, metapath2vec and MetaGraph2Vec along with the homogeneous models, and we attribute this to the usage of the Hadamard function that is able to better capture the relationship between nodes and metapaths than the heterogeneous SkipGram in metapath2vec and MetaGraph2Vec, and the proximity distance mechanism in HINE. M-HIN-path outperforms HIN2Vec on all datasets, which is due to the fact that using the Hadamard function in complex space could help handling complex metapaths being symmetric and asymmetric. Moreover, M-HIN performs even better than M-HIN-path, which could be attributed to two aspects: (1) metagraphs are more expressive than metapaths containing more information between nodes, and (2) complex network embedding is able to deal with complex meta-structures including metapaths and metagraphs.

4.3 Link prediction

In this section, we report on the outcomes of a link prediction task using the node vectors learned by M-HIN. Given a HIN, we first extract a sub-network by selecting an edge class and randomly removing a certain fraction (20%) of the selected edge class as to-be-predicted edges. Then after learning the vector embeddings of the sub-network, we rank node pairs that are more likely to have missing edges in a supervised way. Specifically, we randomly extract 1500 nodes to form a training set and adopted five-fold cross validation. We use Mean Average Precision (MAP) and the top-m recall (R@m) as the evaluation metrics and set m to 100.

Table 3 shows the performance on the link prediction task. Overall, M-HIN outperforms all other models on every dataset regardless of their scale. Unlike in node classification, HINE, metapath2vec and MetaGraph2Vec are consistently better than all homogeneous models DeepWalk, LINE and node2vec; in the link prediction task, metapaths play a more important role than random walks and handle the relationship between nodes more precisely and properly. HIN2Vec still performs better than HINE, metapath2vec and MetaGraph2Vec, but worse than M-HIN-path, which confirms the effectiveness of using the Hadamard function in complex space. M-HIN obtains the best performance among all the models including HIN2Vec and M-HIN-path, demonstrating the advantage of the representation ability of metagraphs and complex embedding.

5 CONCLUSIONS

In this paper, we propose a new model, namely, M-HIN to handle the representation learning issues of a HIN. We utilize nodes and metagraphs between them to construct HIN triplets. Then we apply the Hadamard function to describe the relationship between nodes and metagraphs. In addition, we adopt the complex embedding to deal with the symmetric and asymmetric metagraphs to further improve the model's ability of representation. We compare our model with other baselines on four real-world dataset. M-HIN significantly and consistently outperforms these baselines on two tasks, i.e., node classification and link prediction.

In future work, it is of interest to see whether the emerging graph neural network [1] is applicable to HINs, and how it performs compared with M-HIN.

ACKNOWLEDGMENT

This work was supported by NSFC under grants Nos. 61872446 and 71690233, and NSF of Hunan province under grant No. 2019JJ20024, and by Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Peter W. Battaglia, Razvan Pascanu, and et al. 2016. Interaction Networks for Learning about Objects, Relations and Physics. In *NIPS*. 4502–4510.
[2] Yuxiao Dong, Nitesh V. Chawla, and et al. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *KDD*. 135–144.
[3] Mohammed Elseidy, Ehab Abdelhamid, and et al. 2014. GRAMI: Frequent Subgraph and Pattern Mining in a Single Large Graph. *PVLDB* 7, 7 (2014), 517–528.
[4] Yuan Fang, Wenqing Lin, and et al. 2016. Semantic proximity search on graphs with metagraph-based learning. In *ICDE*. 277–288.
[5] Tao-Yang Fu, Wang-Chien Lee, and et al. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*. 1797–1806.
[6] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *KDD*. 855–864.
[7] Zhipeng Huang and Nikos Mamoulis. 2017. Heterogeneous Information Network Embedding for Meta Path based Proximity. *CoRR* abs/1701.05291 (2017).
[8] Bryan Perozzi, Rami Al-Rfou, and et al. 2014. DeepWalk: online learning of social representations. In *KDD*. 701–710.
[9] Jian Tang, Meng Qu, and et al. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.
[10] Théo Trouillon, Christopher R. Dance, and et al. 2017. Knowledge Graph Completion via Complex Tensor Factorization. *JMLR* 18 (2017), 1–38.
[11] Daokun Zhang, Jie Yin, and et al. 2018. MetaGraph2Vec: Complex Semantic Path Augmented Heterogeneous Network Embedding. In *PAKDD*. 196–208.