RTANet: Recommendation Target-Aware Network Embedding

Qimeng Cao[2], Qing Yin[1], Yunya Song[3], Zhihua Wang[4], Yujun Chen[5], Richard Yi Da Xu[6], Xian Yang[1]*

[1]Alliance Manchester Business School, The University of Manchester, Manchester, United Kingdom
[2]Department of Computer Science, Hong Kong Baptist University, Hong Kong
[3]Department of Journalism, Hong Kong Baptist University, Hong Kong
[4]China Shanghai Institute for Advanced Study of Zhejiang University, Shanghai, China
[5]Recurrent AI, Beijing, China
[6]Department of Mathematics, Hong Kong Baptist University, Hong Kong
csqmcao@comp.hkbu.edu.hk, qingyin9172@gmail.com, yunyasong@hkbu.edu.hk, zhihua.wang@zju.edu.cn, chenyujun@rcrai.com, xuyida@hkbu.edu.hk, xian.yang@manchester.ac.uk

Abstract

Network embedding is a process of encoding nodes into latent vectors by preserving network structure and content information. It is used in various applications, especially in recommender systems. In a social network setting, when recommending new friends to a user, the similarity between the user's embedding and the target friend will be examined. Traditional methods generate user node embedding without considering the recommendation target. No matter which target is to be recommended, the same embedding vector is generated for that particular user. This approach has its limitations. For example, a user can be both a computer scientist and a musician. When recommending music friends with potentially the same taste to him, we are interested in getting his representation that is useful in recommending music friends rather than computer scientists. His corresponding embedding should consider the user's musical features rather than those associated with computer science with the awareness that the recommendation targets are music friends. In order to address this issue, we propose a new framework which we name it as Recommendation Target-Aware Network embedding method (RTANet). Herein, the embedding of each user is no longer fixed to a constant vector, but it can vary according to their specific recommendation target. Concretely, RTANet assigns different attention weights to each neighbour node, allowing us to obtain the user's context information aggregated from its neighbours before transforming this context into its embedding. Different from other graph attention approaches, the attention weights in our work measure the similarity between each user's neighbour node and the target node, which in return generates the target-aware embedding. To demonstrate the effectiveness of our method, we compared RTANet with several state-of-the-art network embedding methods on four real-world datasets and showed that RTANet outperforms other comparative methods in the recommendation tasks.

Introduction

Network embedding, i.e., mapping nodes of a network into latent representations, plays an important role in capturing complex relationships and facilitating downstream analyses, such as node classification [Bhagat, Cormode, and Muthukrishnan 2011; Tang, Aggarwal, and Liu 2016; Zhang, Li, and Gan 2021], link prediction [Murata and Moriyasu 2007; Wang et al. 2015; Daud et al. 2020], and recommendation [Perozzi, Al-Rfou, and Skiena 2014; Shi et al. 2018; Santos, Lelkes, and Levin 2021]. Many network embedding approaches (e.g., [Bojchevski and Günnemann 2017; Perozzi, Al-Rfou, and Skiena 2014; Veličković et al. 2018; Wang et al. 2019; Yang et al. 2015]) assume that each node can be represented by a vector that contains both the node's structural and content information. For example, skip-gram based approaches like DeepWalk or Node2Vec [Grover and Leskovec 2016; Perozzi, Al-Rfou, and Skiena 2014] adopted the idea of word embedding to encode network structural information through random walks, returning fixed embedding vectors for all nodes. Graph neural networks based methods [Fan et al. 2019; Schlichtkrull et al. 2018; Zhou et al. 2020; Chen et al. 2021] encode network data by iteratively aggregating local network neighbourhood information. Graph attention networks (GATs) [Veličković et al. 2018; Busbridge et al. 2019] follow a similar strategy and adopt an attention mechanism to measure impacts from neighbour nodes. Our paper focuses on investigating network embedding methods in the recommendation tasks, in which the similarities of node embeddings need to be examined.

Conventional network embedding methods tend to embed each node to a fixed vector in the recommendation tasks. However, this assumption will not hold in real-world scenarios. For instance, users in a social network will show interest in multiple areas, such as Jonny Greenwood who is both a famous guitarist and a big fan of programming. When recommending a computer scientist to Jonny, Jonny's programming-related characteristics should gain more attention. When recommending musicians who would potentially share common interests with Jonny, we should also consider that Jonny is a world-leading guitarist from a famous rock band. In some works, this was achieved by introducing pairwise edges representation [Tu et al. 2017], or edge-aware nodes representation [Deng et al. 2021]. However, these methods have high space complexity, which prevents them from wider applications. Different from these works, our work sug-

*The corresponding author.

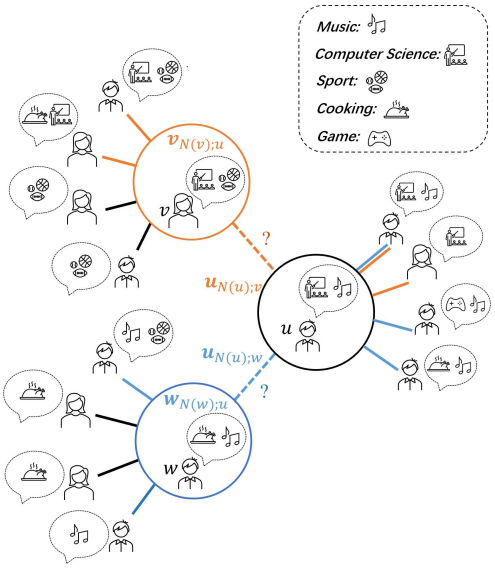


Figure 1: An illustration of RTANet for friend recommendations in a social network. For a given node u, when considering whether node v shall be recommended or not, our RTANet model will first collect neighbour nodes of u, denoted as $N(u)$. Then, it will assign weights to nodes in $N(u)$ via an attention mechanism. The friends of u having similar interests with v would gain more attention (indicated by orange lines) and contribute more in generating $\boldsymbol{u}_{N(u);v}$. If we are interested in recommending a different node w to u, then a different embedding $\boldsymbol{u}_{N(u);w}$ will be generated, where the friends of u having common interests with w gain more attention (indicated by blue lines). Similarly, node v and w will also get their embeddings $\boldsymbol{v}_{N(v);u}$ and $\boldsymbol{w}_{N(w);u}$ in the recommendation tasks.

gests learning the context information from neighbour nodes aggregated with the awareness of the recommendation target will significantly reduce storage space requirements.

As mentioned above, we develop a recommendation target-aware network embedding method (RTANet) to assist recommendations.

Specifically, as depicted in Figure 1, in the task of recommending node v to node u, u's target-aware embedding $\boldsymbol{u}_{N(u);v}$ contains its context information from its neighbour nodes $N(u)$. The target-aware embedding $\boldsymbol{u}_{N(u);v}$ is obtained via the conditional representation module based on the attention mechanism. The attention weights of $N(u)$ will be influenced by the recommendation target v. When recommending another node w to u, the weights would vary since they depend on the similarities between u's neighbour nodes and the target node w. As depicted in Figure 2, the attention mechanism adopted in RTANet is different from the conventional attention mechanism as used in the graph attention networks (GATs) [Veličković et al. 2018; Busbridge et al. 2019]. In GAT, the self-attention mechanism will assign attention weights to node u's neighbours depending on their similarities with u itself and return a fixed embedding $\boldsymbol{u}_{N(u)}$, while in RTANet the attention weights over node u's neighbours rely on the recommendation target.

Our contributions can be summarized as follows:

- We propose a recommendation target-aware network embedding approach, in which the embedding of a node in a network is not fixed but can be variable depending on the recommendation target.
- Our model can deal with both heterogeneous and homogeneous networks. In the experiment, we have shown the state-of-the-art performance on four representative real-world datasets from two social networks, one citation network, and one clinical network.

Related Work

Learning Fixed Node Embeddings

Network embedding maps nodes in the network into a latent space by preserving their structural and content information. The learned embedding vectors can be applied to a variety of downstream tasks including recommendation (e.g., [Dong, Chawla, and Swami 2017; Fan et al. 2018; Shang et al. 2016; Shi et al. 2018; Sun et al. 2018]), which is our main interest. Currently, most of the network embedding studies focus on learning the embeddings by extracting network information in an unsupervised manner. For example, DeepWalk [Perozzi, Al-Rfou, and Skiena 2014] borrowed the idea from word embedding to learn the node embeddings. Many methods followed a standard pipeline and only considered structural information [Grover and Leskovec 2016; Ribeiro, Saverese, and Figueiredo 2017; Tang et al. 2015], while some other methods like TADW [Yang et al. 2015], GAT [Veličković et al. 2018; Wang et al. 2019], and Graph2Gauss [Bojchevski and Günnemann 2018] considered both network structural information and content information. Among them, CNE [Kang, Lijffijt, and De Bie 2018] proposed a conditional network embedding approach to learn node embeddings conditioned on topological properties such as node degrees and block densities. CSNE [Mara et al. 2020] adopted a similar idea to conditional embeddings in signed networks. All these methods focus on generating a fixed embedding vector for each node in a network, which cannot reflect the diversity of interests in recommendations.

Learning Variable Node Embeddings in Recommendation Tasks

In the domain of network embedding, context-aware learning is also being developed, in which a variable embedding of each node is generated based on the varying context. The CANE method, proposed in [Tu et al. 2017], encodes the information of each node in the network into two vectors, which are the structure-based embedding and text-based embedding. The mutual attention mechanism is used to obtain context-aware text embedding. Based on this work, the studies in [Tian et al. 2018; Shen et al. 2018; Chen et al. 2019b] proposed a word alignment based network embedding method to match significantly relevant words between two text pieces

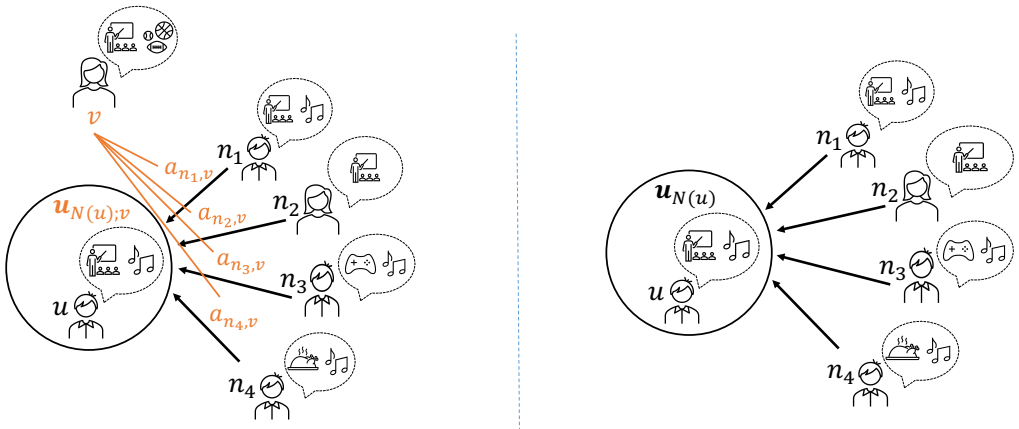


Figure 2: Comparison between attention mechanisms in RTANet and GAT. In RTANet, the importance of each u's neighbour node n_i (i.e. $a_{n_i,v}$) depends on the attention between v and n_i, where v is the node to be recommended. Therefore, choosing different v for recommendation would result in different embeddings of u. In a standard GAT, the importance of n_i is not related to v, generating a fixed embedding vector.

from paired nodes. The work in [Wu et al. 2019] further extended the idea of context-aware network embedding for item recommendation by introducing an attraction layer for semantic feature learning. Similarly, the CAME method developed in [Wang et al. 2020] focused on the content and context-aware music embedding for personalized music recommendation. The content refers to auxiliary music information such as tags, descriptions, and lyrics, while the context refers to user nodes and other related music in the playlist. The work in [Yu, Wanyan, and Wang 2021] leveraged context information to enhance the point-of-interest recommendation. A deep relational learning framework was developed in [Pacheco and Goldwasser 2021] to model the sentence similarity for natural languages tasks, where each sentence can be formulated as a node under its framework.

The learning outputs of these methods can be summarized as follows: for recommending node v to node u, they return the pairwise conditional embeddings $\boldsymbol{u}_v$ and $\boldsymbol{v}_u$, denoting the embedding vectors of node u with given v, and node v with given u, respectively. Their learning objective can be decomposed into the structure objective and the content objective. These approaches have to store all the embeddings for different edges, and would require extremely large storage space if the network is huge. In our RTANet model, we use the conditional representation module to learn variable node embeddings, where a different attention scheme is developed and the storage can be saved greatly.

Materials and Methods

Problem Definition

Definition 1. Network Embedding
Network Embedding can be formulated as follows: given a network $G = (V, E)$, where V is a set of nodes and $E \subseteq V \times V$ is a set of links between nodes, the task is to learn a mapping function $f : V_i \rightarrow \mathbf{v}_i \in \mathbb{R}^d$. The output $\mathbf{v}_i$ is a multidimensional embedding vector, where the dimension of embedding space d satisfies $d \ll |V|$. The node embeddings are learned by preserving the network structure and content information. Conventional network embedding methods merely return a fixed embedding vector for each node that would not change in different tasks. The resulting embedding is denoted as $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{|V|})' \in \mathbb{R}^{|V| \times d}$.

Definition 2. Recommendation Target-Aware Network Embedding
Relational data can be characterized as a network $G = (V, E)$, where V and E have the same definitions as above. For a given node u, deciding whether to recommend a target node v to u depends on the similarity between their embeddings. We assume that the embedding of u in this recommendation task contains two kinds of information: the static information $\mathbf{u}$ encoding its inherent properties, and the context information from its neighbour nodes, i.e., $N(u)$. The goal of a recommendation target-aware network embedding is to find a mapping function $f : u \rightarrow \mathbf{u}_{N(u);v} \in \mathbb{R}^d$, where $\mathbf{u}_{N(u);v}$ refers to the embedding of node u containing the context information from $N(u)$ in the task of recommending node v. Similarly, the embedding of node v can be denoted by $\mathbf{v}_{N(v);u}$ when we want to recommend u to v. The similarity between $\mathbf{u}_{N(u);v}$ and $\mathbf{v}_{N(v);u}$ will be evaluated by distance metrics, such as the cosine distance.

Our Method

Figure 3 gives an overview of our proposed RTANet model by taking the triplet (u, v, v') as a training sample, where node u is linked to node v but not linked to node v' in a network. The embedding vectors $\mathbf{u}_{N(u);v}$, $\mathbf{u}_{N(u);v'}$, $\mathbf{v}_{N(v);u}$ and $\mathbf{v'}_{N(v');u}$ are generated by the conditional representation learning module. The triplet loss is calculated using the

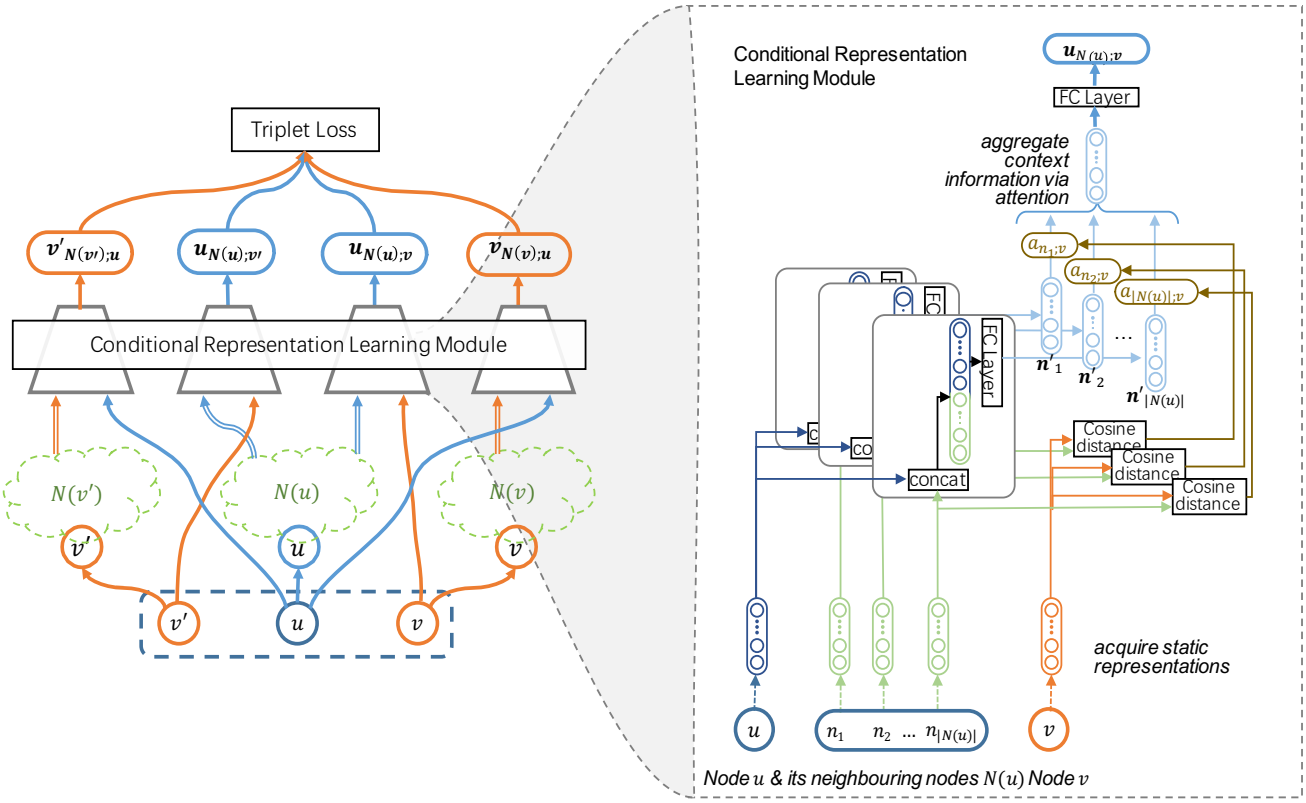


Figure 3: *Left*: An illustration of proposed recommendation target-aware network embedding model (RTANet). An instance of the input is represented by a triplet (u, v, v'), where node u and v are linked (i.e. positive pair) and u and v' are not linked (i.e. negative pair) in the given network. Their embeddings are generated via the conditional representation learning module. Triplet loss is designed to make the distance between node embeddings in the negative pair larger than the distance for the positive pair. *Right*: An illustration of the conditional representation learning module for generating $\mathbf{u}_{N(u);v}$.

outputs of the conditional representation learning module. The subplot on the right of Figure 3 depicts this conditional module for generating $\mathbf{u}_{N(u);v}$, whose inputs are u, v and $N(u)$ (i.e. u's neighbour nodes). For each neighbour node $n_i \in N(u)$, its static embedding is concatenated with the static vector $\mathbf{u}$ and then fed into a fully connected (FC) layer. The outputs of the FC layer, $\{\mathbf{n_i}'\}_{i=1}^{|N(u)|}$, are aggregated into one single vector $\mathbf{u}_{N(u);v}$ via attention. Each attention weight value $a_{n_i,v}$ assigns importance to each $\mathbf{n}'_i$ by calculating the cosine distance between $\mathbf{v}$ and $\mathbf{n}_i$. In the following part, we will first introduce details of the conditional representation learning module for generating embeddings via attention. Then we will define the objective learning function based on the triplet loss. Some key notations used in this section are listed in Table 1.

The Conditional Representation Learning Module The conditional representation learning module is designed to generate the target-aware node embeddings in the recommendation tasks. An example workflow of generating the embedding $\mathbf{u}_{N(u);v}$ in recommending a node v to u is illustrated in the right subplot of Figure 3. Let $\mathbf{n}_i \in \mathbb{R}^d$ be the static embedding of any node $n_i \in N(u)$ and $\mathbf{v} \in \mathbb{R}^d$ be the static node embedding v. These embeddings would be learned during the model training process. The weight vector $\mathbf{a}_{N(u);v} \in \mathbb{R}^{|N(u)|}$, indicating different contributions of nodes in $N(u)$ to get $\mathbf{u}_{N(u);v}$, is obtained via the attention mechanism as:

$$\begin{aligned}\mathbf{a}_{N(u);v} &= softmax([a_{n_1,v}, a_{n_2,v}, ..., a_{n_{|N(u)|},v}]), \\ &= \left[\frac{e^{a_{n_1,v}}}{\sum_{n_i \in N(u)} e^{a_{n_i,v}}}, ..., \frac{e^{a_{n_{|N(u)|},v}}}{\sum_{n_i \in N(u)} e^{a_{n_i,v}}}\right] \end{aligned} \quad (1)$$

and

$$a_{n_i,v} = Cosine(\mathbf{n}_i, \mathbf{v}) = \frac{\mathbf{n}_i \cdot \mathbf{v}}{||\mathbf{n}_i|| ||\mathbf{v}||}, \quad (2)$$

where $||\cdot||$ returns magnitude of a vector. Each weight value is the cosine distance between a neighbour node n_i and v; and *softmax* is used for normalization.

Instead of directly using the weighted embedding $\sum_i a_{n_i,v} \cdot \mathbf{n}_i$, we also include the static information $\mathbf{u} \in \mathbb{R}^d$ into the target-aware embedding of u. Here, we combine each $\mathbf{n}_i$ with $\mathbf{u}$ via neural networks:

$$\mathbf{n'}_i = FC_1([\mathbf{n}_i \oplus \mathbf{u}]; \boldsymbol{\theta_1}]) \quad (3)$$

where $\mathbf{n'}_i \in \mathbb{R}^d$ is the combined embedding, $\boldsymbol{\theta}_1$ refers to the parameters of the fully-connected layer with non-linear acti-

Symbols	Descriptions
u	A node in the network.
v	A node being linked to u.
v'	A node not being linked to u.
(u, v, v')	A triplet sample.
T	A dataset containing triplet samples.
$N(u)$	All first-order neighbours of node u.
n_i	A node in $N(u)$, $i \in \{1, 2, 3, ..., \lvert N(u)\rvert\}$.
d	The embedding size.
r	The negative sampling ratio used in the contrastive training strategy.
m	The margin parameter in the loss.
$\mathbf{u}_{N(u);v}$	The target-aware embedding of node u containing the context information from $N(u)$ in the task of recommending node v.
$\mathbf{a}_{N(u);v}$	The attention vector used to derive $\mathbf{u}_{N(u);v}$.
$\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$	Parameters of neural networks.
$\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{v'}, \mathbf{n}_i$	The static embeddings.
$FC_1(\cdot), FC_2(\cdot)$	Fully connected neural networks.

Table 1: Symbols and Descriptions.

vation $FC_1(\cdot)$, and the symbol $\oplus$ stands for the concatenation operation for two vectors.

Having obtained the combined embedding $\mathbf{n'}_i$ for all $n_i \in N(u)$, the following step is to aggregate them into $\mathbf{u}_{N(u);v}$ using the weight vector $\mathbf{a}_{N(u);v}$ by:

$$\mathbf{u}_{N(u);v} = FC_2(\sum_{i=1}^{|N(u)|} a_{n_i,v} \cdot \mathbf{n'}_i; \boldsymbol{\theta}_2), \tag{4}$$

where $FC_2(\cdot; \boldsymbol{\theta}_2)$ represents a fully-connected layer with an activation function parameterized by $\boldsymbol{\theta}_2$. In practice, *Sigmoid* and *ReLU* are the most effective and frequently used activation functions.

Learning Objective Function Many network embedding methods construct objective functions by comparing node pairs through a pair-wise learning manner, like skip-gram and metric learning. For example, in DeepWalk [Perozzi, Al-Rfou, and Skiena 2014] and Node2Vec [Grover and Leskovec 2016], the embedding of any node $u \in V$ was obtained through a skip-gram model, aiming at maximizing the probability of obtaining its neighbour nodes $N(u) \subset V$ in the following random-walk sequence:

$$\max_f \sum_{u \in V} \log Pr(N(u)|f(u)), \tag{5}$$

where $f(\cdot)$ is the embedding function. In order to make the optimization tractable, we give a common assumption of conditional independence. The likelihood can be factorized as:

$$Pr(N(u)|f(u)) = \prod_{n_i \in N(u)} Pr(n_i|f(u)). \tag{6}$$

The result of optimization is the learned mapping function $f(\cdot)$, with which embeddings of all nodes (i.e., $f(u) \in \mathbb{R}^d$ for all $u \in V$) in the network can be obtained. Then for recommendation, different metrics such as consine distance can be used to measure the relevance of two nodes via their embeddings.

In this paper, our model adopts a contrastive training strategy [Chen et al. 2020]. For a given node u, we include a pair of contrastive nodes v and v' to form a triplet (u, v, v'), where v is a positive node (i.e., there is a link between u and v) and v' is a negative node (i.e., v is not linked to u). The reason for including negative nodes is that if our loss is purely related to positive nodes, the learning process would be intractable. In the training process, we randomly sample triplets with the negative sampling ratio r.

To calculate the training loss, we first generate the embedding vectors $\mathbf{u}_{N(u);v}$ and $\mathbf{v}_{N(v);u}$ using Eq. 4 for the positive node pair (u, v). Similarly, for the negative node pair (u, v'), we generate the embedding vectors $\mathbf{u}_{N(u);v'}$ and $\mathbf{v'}_{N(v');u}$. With these vectors, we then choose the triplet loss to implement the recommendation task. Triplet loss is widely used in various recommendation scenarios and has shown great performance. It defines the learning objective in such a way that the distance between nodes in the negative pair is larger (bigger than a margin m) than the distance between nodes in the positive pair, that is:

$$\begin{aligned}\mathcal{L} = \sum_{(u,v,v') \in T} max(0, m + d(\mathbf{u}_{N(u);v}, \mathbf{v}_{N(v);u}) \\ - d(\mathbf{u}_{N(u);v'}, \mathbf{v'}_{N(v');u})),\end{aligned} \tag{7}$$

where $d(.)$ is the distance function which is chosen to be cosine in our model, and T denotes the whole dataset containing triplet samples. This loss can be efficiently minimized by stochastic gradient descent (SGD) through sampling mini-batches from the training set. Algorithm 1 further describes the training process.

Algorithm 1: Recommendation Target-Aware Network Embedding

Input: A network $G = (V, E)$, where V is a set of nodes and $E \subseteq V \times V$ is a set of links between nodes.
Output: The embedding function $f : u \rightarrow \mathbf{u}_{N(u);v} \in \mathbb{R}^d$ for $\forall u \in V$.
Initialize: $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, and $\mathbf{u}$ for $\forall u \in V$.
not converge Select a node u;
Sample a positive node v and a negative node v' from the network G to get a triplet (u, v, v');
Collect the neighbour nodes $N(u)$, $N(v)$ and $N(v')$;
Calculate $\mathbf{u}_{N(u);v}$, $\mathbf{v}_{N(v);u}$, $\mathbf{u}_{N(u);v'}$, $\mathbf{v'}_{N(v');u}$ using Equation 4;
Calculate the loss $\mathcal{L}$ using Equation 7;
Update model parameters $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ and static embeddings of nodes via backward propagation.

	Flickr		Yelp		Author-Dataset		MIMIC-III	
	Name	**Number**	**Name**	**Number**	**Name**	**Number**	**Name**	**Number**
Node	User Node	7,573	User Node	14,031	Author Node	73,855	Patient Node	58,929
			Business Node	13,938	Paper Node	79,775	Disease Node	6,983
			Category Node	575	Dataset Node	6,863	Drug Node	2,048
Link	Relation Link	239,738	Business-Category Link	39,406	Author-Paper Link	191,096	Patient-Drug Link	1,510,689
			Business-User Link	194,236	Paper-Dataset Link	15,572	Patient-Disease Link	650,932
Attributes	Average Degree	63.32	Average Degree	16.37	Average Degree	2.58	Average Degree	63.61
	Assortative Coefficients	-0.2275	Assortative Coefficients	-0.0566	Assortative Coefficients	-0.0135	Assortative Coefficients	-0.3076

Table 2: Summary statistics of four datasets.

Computational Complexity In the training phase, we adopt the mini-batch gradient descent approach to infer the parameters of our model. We first sample a batch of triplets (u, v, l) (l represents a positive or negative label) and then take a gradient step to optimize the loss function as stated in Equation 7. As a result, the time complexity is only related to the maximum degree of nodes and the number of nodes $|V|$ in the network. In the inference phase, the computational complexity of learning embeddings for each node only depends on its number of neighbours. As a result, our RTANet method can be easily scaled to large-size networks, where the computational cost is only linear to the number of nodes $|V|$ in the network.

Experiments

Dataset

In order to comprehensively evaluate the effectiveness of the proposed RTANet model, we utilized four datasets, from two social networks, one citation network and one clinical network. The summary statistics are shown in Table 2 and the detailed descriptions are given as follows.

- **The Social Network Dataset (Flickr)** [Huang, Li, and Hu 2017] The Flickr dataset is collected from a social network, where nodes represent users and links refer to connections among users. In total, there are 7,573 user nodes with 239,738 links among them.
- **The Yelp Dataset** [Zheng et al. 2016] The Yelp dataset is collected from the famous user review website, Yelp, which contains the connections between businesses and users as well as relationships among users. Here, we only focus on the business-user links and business-category links. In total, there are 14,031 user nodes, 13,938 business nodes and 575 category nodes with 39,406 business-category links and 194,236 business-user links.
- **The Author-Dataset Dataset** [Yujun et al. 2019] The Author-Dataset dataset is from a citation network containing nodes of three types, which are authors, papers and datasets from the domain of machine learning, data mining, and computer vision. In our experiment, there are 73,855 author nodes, 79,775 paper nodes, and 6,863 dataset nodes with 191,096 author-paper links and 15,572 paper-dataset links.
- **The MIMIC-III Dataset** [Johnson et al. 2016] We use electronic health record (EHR) data from MIMIC-III, in which each record contains detailed information of one hospital visit for a patient. Here, we extract drug codes (DrugBank [Wishart et al. 2008]) and disease codes (ICD [Wishart et al. 2008]) from each EHR and use them to construct a clinical network. In this network, nodes represent patients, diseases, and drugs, while links indicate their co-occurrence. In total, there are 67,960 nodes (patients:58,929, diseases: 6,983, drugs: 2,048) and 2,161,621 links (patient-drug link: 1,510,689, patient-disease link: 650,932).

Comparative Methods

To better evaluate the proposed method, we compare RTANet with four state-of-the-art recommendation methods, which are listed as follows:

- **DeepWalk** [Perozzi, Al-Rfou, and Skiena 2014]: DeepWalk is a representative network embedding method for general usage. It only uses network structural information to generate a fixed multidimensional embedding vector.
- **Node2Vec** [Chen et al. 2019a]: Node2Vec is also a popular network embedding method that generates structure-based node embeddings. This method adopts a sampling strategy different from DeepWalk on the purpose of better capturing both the global and local structural properties. The outputs of this method are also fixed node embeddings.
- **CNE** [Kang, Lijffijt, and De Bie 2018]: Conditional network embedding (CNE) is a conditional network embedding method. However, the node embeddings are only conditioned on their degrees. Similar to the methods above, this method also generates fixed node embeddings for all downstream tasks.
- **CANE** [Tu et al. 2017]: CANE is a context-aware network embedding method, which learns various embeddings of a node according to its varying context.

Experimental Settings We implement all baseline methods based on the source codes provided by authors. Hyperparameters of all baselines are set as described in the original papers. To achieve a fair comparison, we set the dimension of embedding vectors (i.e., d) of all methods to be 256. For CANE, we used the default values of α, β, and γ and set

Datasets	Percentage	DeepWalk		Node2Vec		CNE		CANE		RTANet	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
Flickr	50%	0.57	0.575	0.575	0.581	0.602	0.612	0.633*	0.623*	**0.685**	**0.694**
	20%	0.592	0.581	0.6	0.59	0.609	0.611	0.651*	0.650*	**0.706**	**0.702**
	5%	0.608	0.601	0.612	0.609	0.613	0.618	0.659*	0.655*	**0.718**	**0.709**
Yelp	50%	0.786	0.778	0.789*	0.781 *	0.750	0.758	0.765	0.760	**0.830**	**0.823**
	20%	0.795	0.789	0.798	0.794	0.805	0.803	0.811*	0.815*	**0.853**	**0.851**
	5%	0.809	0.803	0.810	0.803	0.808	0.807	0.828*	0.824*	**0.882**	**0.891**
Author-Dataset	50%	0.852	0.881	0.857	0.886	0.873	0.889	0.924*	0.916*	**0.966**	**0.961**
	20%	0.919	0.925	0.924	0.929	0.935	0.948*	0.945*	0.937	**0.971**	**0.965**
	5%	0.936	0.937	0.941	0.937	0.947	0.953*	0.957*	0.942	**0.975**	**0.970**
MIMIC-III	50%	0.729	0.711	0.742	0.727	0.811*	0.805*	0.805	0.799	**0.862**	**0.849**
	20%	0.733	0.725	0.767	0.759	0.832	0.829*	0.839*	0.820	**0.895**	**0.887**
	5%	0.772	0.758	0.791	0.774	0.841	0.833*	0.850*	0.833*	**0.904**	**0.901**

Table 3: The performance of different network embedding methods (the best results are in bold, while the second best ones are labelled with $*$). The percentage values indicate the proportions of links used as testing samples.

negative sampling parameter k to be 1. In the implementation of our RTANet model, the negative sampling ratio r is also set to be 1, while the margin m in triplet loss is set to 1 according to empirical results. The hidden size of attention module is set as 256, which is the same as other baselines. We train our model through five epochs by default and also adopt the early stopping strategy. The focus of the experiments is to learn node embeddings by encoding network structural information. Our method has not been compared with the network embedding methods which also encode the node content information such as texts and images. However, our proposed method can be naturally extended to encode content information.

Quantitative Results

In this section, we evaluate the effectiveness of RTANet in the recommendation tasks. Here, we formulate the problem as the pair-wise node recommendation.

For Flickr, we predict user-user links in the friend recommendation tasks, where the context is from the information of neighbouring user nodes; for Yelp, we predict user-business links in the shop recommendation tasks, where the context is from the information of categories of user interests; for the Author-Dataset dataset, we predict author-dataset links in the dataset recommendation tasks, where the context is from the information of neighbouring paper nodes; for MIMIC-III, we predict patient-drug links in the drug recommendation tasks using the context information from patients' neighbouring disease nodes.

For all datasets, we evaluate the AUC (Area Under ROC Curve) and AP (Average Precision) values by removing multiple links from the original networks at different ratios. Here, we create three test sets containing 50%, 20% and 5% randomly selected links for performance evaluation. Table 3 shows the results of different comparative methods. The observations are summarized as follows:

- Our model, RTANet, outperforms all baseline methods with significantly higher AUC and AP values for all datasets at different testing ratios.
- The conditional network representation learning methods CNE and CANE show better performance than conventional DeepWalk and Node2Vec methods for most datasets.
- Compared with CANE which also generates variable node embeddings, our RTANet method returns higher AUC and AP values. This is because CANE only considers one node as a context node, while RTANet generates target-aware embeddings by aggregating context information from multiple neighbour nodes.
- The performance of all methods would be improved as the value of the testing ratio decreases. However, the performance of our model, RTANet, at the largest testing ratio (i.e. 50%) is better than most of the other baselines at the smallest testing ratio (i.e. 5%). It indicates that our model is more robust than the others with limited training samples.

Case Studies

In this subsection, we conduct two case studies to demonstrate the ability of RTANet for generating multiple embeddings of the same node in different recommendation tasks. The t-SNE method [Van der Maaten and Hinton 2008] is used to visualize embedding vectors.

Figure 4 illustrates the first case study from the Yelp Dataset, where the example local business belongs to two categories: 'breakfast' and 'restaurant'. In this case, we can see that this local business has two different embeddings encoding information corresponding to two categories. Meanwhile, we find that the embeddings of users related to this local business form two different clusters. One cluster contains users that are likely to comment on the breakfast, while the other cluster has users who are happy to share their experiences about the dinner. Notably, the business embeddings are dragged towards the direction of related users. This case shows that RTANet can generate informative embeddings in the restaurant recommendation task.

Figure 5 shows a more complex case, where multiple em-

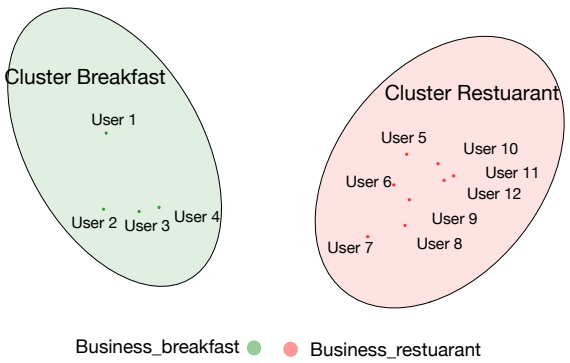


Figure 4: The t-SNE visualization for the first case study. We select one local business from the Yelp dataset which belongs to two categories: 'breakfast' and 'restaurant'. Meanwhile, there are many users commenting on this business, whose embeddings are also visualized here. The embeddings of these users form two clusters. The business embeddings are dragged towards the direction of related user clusters.

beddings of an example patient node from the MIMIC-III dataset are generated in different drug recommendation tasks. In this case study, 7 different drugs are recommended to the same patient: 'Irbesartan' and 'Eplerenone' treat similar diseases like hypertension and heart failure; 'Morphine' and 'Nitroglycerin' are for pain relief; 'Zonisamide' is to treat seizures; 'Dextrose' and 'Potassium chloride' are drugs for metabolic diseases. For each drug, the context information from neighbouring disease nodes would be different, generating 7 embedding vectors of this patient. From Figure 5, we can see that the patient embeddings are dragged towards the direction of related drugs. Some embedding vectors of the patient are located closer than the others (on the left part of Figure 5). For example, embeddings of the same patient resulting from recommending 'Morphine' and 'Nitroglycerin' nearly overlap with each other. These two drugs are both pain killers and their own embeddings are located closely as well (on the right part of Figure 5). Similarly, if two embedding vectors of the patient are located far away from each other, their corresponding drugs' embeddings would also be quite different (e.g., 'Irbesartan' and 'Potassium chloride').

Sensitivity of Hyperparameters

One primary hyperparameter of RTANet is the embedding size d. The above experimental results are all from the same setting, where $d = 256$. Figure 6 (a) shows the performance of RTANet when d is chosen from $\{2, 4, 8, 16, 32, 64, 128, 256, 512\}$ for the MIMIC-III dataset with 20% testing links. By looking at the changes in AUC over different values of d, we can see that when d is larger than 16 the performance of RTANet is stable in terms of prediction power. However, even for values as low as $d = 2$, the AUC values can still reach around 83%. These observations indicate the stability of RTANet on different values of the hyperperameter d.

Another hyperparameter we have investigated is the negative sampling ratio r, introduced in the contrastive training. This value determines the number of triplets used for training. In the previous experiments, r is set to 1. Figure 6 (b) explores the impacts of different values of $r \in \{0.1, 0.5, 1, 2, 10\}$ on the MIMIC-III dataset with 20% testing links. It is found that the AUC values are quite consistent. Even when the negative sampling ratio is as low as 0.1, the AUC value is still as high as 0.88. These observations show that our model does not need to randomly generate many negative samples in the training process, which is quite data efficient.

Ablation Study on Loss Function

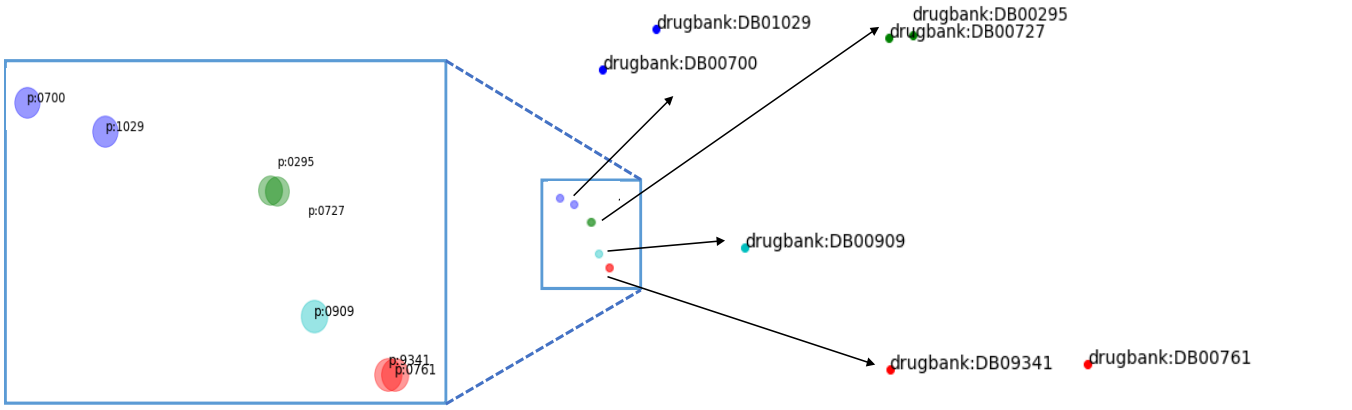
In our experiment, we also study the impacts of different loss functions. We conduct the ablation study on the Yelp dataset with only 5% links being masked as testing samples. Here, we compare our triplet loss with the Cross Entropy loss, which is commonly used in other baselines like Node2Vec and CNE. Table 4 shows that the resulted AUC and AP scores are quite similar, where triplet loss has obtained slightly higher values. This result indicates that the triplet loss does not play a key role in performance improvement. It is more likely that the attention mechanism of our RTANet model boosts the performance.

	AUC	AP
RTANet with Triplet Loss	**0.904**	**0.901**
RTANet with Cross Entropy Loss	0.897	0.894

Table 4: Ablation study on the Yelp dataset with 5% links being used as testing samples.

Conclusion

In this paper, we propose a novel network embedding method, RTANet, to facilitate recommendation tasks with improved performance. Unlike conventional network embedding learning methods, our work suggests that the embedding of a node in the network should vary according to the recommendation target. To achieve this goal, we introduce a conditional representation module into RTANet, which adopts an attention mechanism to aggregate context information from the neighbourhood. The attention weight of each neighbour node depends on its similarity with the recommendation target. For model training, we adopt the contrastive training strategy and choose the triplet loss as the learning objective. To demonstrate the effectiveness of our method, RTANet is compared with some of the state-of-the-art network embedding methods and has been evaluated on four real-world datasets. The method proposed in this paper can potentially discover the new paradigm in the field of network embedding. We will investigate the effectiveness of RTANet on other types of networks such as attributed networks and signed networks. An interesting direction of the future development of RTANet would be taking advantage of the attention mechanism to further improve the interpretability of network embedding. Moreover, extending the model to other tasks, such as node classification and community detection, would also be inter-



DrugBank ID	Drug Name	Descriptions from DrugBank
DB01029	Irbesartan	Irbesartan is an angiotensin receptor blocker used to treat hypertension, delay progression of diabetic nephropathy, and treat congestive heart failure.
DB00700	Eplerenone	Eplerenone is an aldosterone receptor antagonist used to improve survival of patients with symptomatic heart failure and to reduce blood pressure.
DB00295	Morphine	Morphine is used for the management of chronic, moderate to severe pain
DB00727	Nitroglycerin	Nitroglycerin is a vasodilator drug used for the treatment of chest pain and high blood pressure.
DB00909	Zonisamide	Zonisamide is a sulfonamide anticonvulsant used to treat partial seizures.
DB09341	Dextrose	Glucose pharmaceutical formulations (oral tablets, injections) are indicated for caloric supply and carbohydrate supplementation in case of nutrient deprivation. It is also used in metabolic disorders such as hypoglycemia.
DB00761	Potassium chloride	For use as an electrolyte replenisher and in the treatment of hypokalemia.

Figure 5: The t-SNE visualization for the second case study. A patient is randomly selected from the MIMIC-III dataset, who is linked with 7 different drugs. The embeddings of these 7 drugs and the corresponding 7 different embeddings of this patient are visualized by t-SNE. For instance, the location of the node labelled with 'drugbank:DB00909' corresponds to the low dimensional representation after t-SNE; the location of 'p:0909' corresponds to the representation of the patient when 'drugbank:DB00909' is recommended.

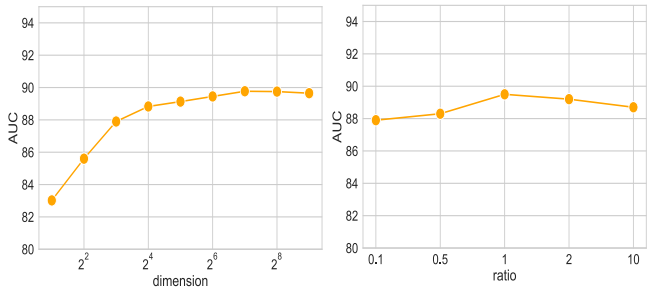


Figure 6: Sensitivity analysis: AUC scores of RTANet for (a) different values of embedding size d, and (b) different values of negative sampling ratio r.

esting from the application perspective. In the future, we will also study the different behaviours of the attention mechanism for homogeneous and heterogeneous networks.

References

Bhagat, S.; Cormode, G.; and Muthukrishnan, S. 2011. Node classification in social networks. In *Social network data analytics*, 115–148. Springer.

Bojchevski, A.; and Günnemann, S. 2017. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*.

Bojchevski, A.; and Günnemann, S. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *ICLR*.

Busbridge, D.; Sherburn, D.; Cavallo, P.; and Hammerla, N. Y. 2019. Relational graph attention networks. *arXiv preprint arXiv:1904.05811*.

Chen, C.; Li, K.; Wei, W.; Zhou, J. T.; and Zeng, Z. 2021. Hierarchical graph neural networks for few-shot learning. *IEEE Transactions on Circuits and Systems for Video Technology*.

Chen, J.; Wu, Y.; Fan, L.; Lin, X.; Zheng, H.; Yu, S.; and Xuan, Q. 2019a. N2VSCDNNR: A Local Recommender System Based on Node2vec and Rich Information Network. *IEEE Transactions on Computational Social Systems*.

Chen, L.; Wang, G.; Tao, C.; Shen, D.; Cheng, P.; Zhang, X.; Wang, W.; Zhang, Y.; and Carin, L. 2019b. Improving textual network embedding with global attention via optimal transport. *arXiv preprint arXiv:1906.01840*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Daud, N. N.; Ab Hamid, S. H.; Saadoon, M.; Sahran, F.; and Anuar, N. B. 2020. Applications of link prediction in

social networks: A review. *Journal of Network and Computer Applications*, 166: 102716.

Deng, J.; Wan, S.; Wang, X.; Tu, E.; Huang, X.; Yang, J.; and Gong, C. 2021. Edge-Aware Graph Attention Network for Ratio of Edge-User Estimation in Mobile Networks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 9988–9995. IEEE.

Dong, Y.; Chawla, N. V.; and Swami, A. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*, 135–144. ACM.

Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; and Yin, D. 2019. Graph neural networks for social recommendation. In *The World Wide Web Conference*, 417–426.

Fan, Y.; Hou, S.; Zhang, Y.; Ye, Y.; and Abdulhayoglu, M. 2018. Gotcha-sly malware!: Scorpion a metagraph2vec based malware detection system. In *KDD*, 253–262. ACM.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *KDD*, 855–864. ACM.

Huang, X.; Li, J.; and Hu, X. 2017. Label informed attributed network embedding. In *WSDM*, 731–739. ACM.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Li-Wei, H. L.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L. A.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1): 1–9.

Kang, B.; Lijffijt, J.; and De Bie, T. 2018. Conditional network embeddings. *arXiv preprint arXiv:1805.07544*.

Mara, A.; Mashayekhi, Y.; Lijffijt, J.; and De Bie, T. 2020. CSNE: Conditional Signed Network Embedding. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1105–1114.

Murata, T.; and Moriyasu, S. 2007. Link prediction of social networks based on weighted proximity measures. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI'07)*, 85–88. IEEE.

Pacheco, M. L.; and Goldwasser, D. 2021. Modeling content and context with deep relational learning. *Transactions of the Association for Computational Linguistics*, 9: 100–119.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710.

Ribeiro, L. F.; Saverese, P. H.; and Figueiredo, D. R. 2017. struc2vec: Learning node representations from structural identity. In *KDD*, 385–394. ACM.

Santos, F. P.; Lelkes, Y.; and Levin, S. A. 2021. Link recommendation algorithms and dynamics of polarization in online social networks. *Proceedings of the National Academy of Sciences*, 118(50).

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, 593–607. Springer.

Shang, J.; Qu, M.; Liu, J.; Kaplan, L. M.; Han, J.; and Peng, J. 2016. Meta-path guided embedding for similarity search in large-scale heterogeneous information networks. *arXiv preprint arXiv:1610.09769*.

Shen, D.; Zhang, X.; Henao, R.; and Carin, L. 2018. Improved semantic-aware network embedding with fine-grained word alignment. *arXiv preprint arXiv:1808.09633*.

Shi, C.; Hu, B.; Zhao, W. X.; and Philip, S. Y. 2018. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31(2): 357–370.

Sun, L.; He, L.; Huang, Z.; Cao, B.; Xia, C.; Wei, X.; and Philip, S. Y. 2018. Joint embedding of meta-path and meta-graph for heterogeneous information networks. In *2018 IEEE International Conference on Big Knowledge (ICBK)*, 131–138. IEEE.

Tang, J.; Aggarwal, C.; and Liu, H. 2016. Node classification in signed social networks. In *Proceedings of the 2016 SIAM international conference on data mining*, 54–62. SIAM.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, 1067–1077. International World Wide Web Conferences Steering Committee.

Tian, L.; Zhang, D.; Han, F.; Hong, M.; Huang, X.; Chen, Y.; and Wu, Y. 2018. Context-Aware Network Embedding via Variation Autoencoders for Link Prediction. In *International Conference of Pioneering Computer Scientists, Engineers and Educators*, 322–331. Springer.

Tu, C.; Liu, H.; Liu, Z.; and Sun, M. 2017. Cane: Context-aware network embedding for relation modeling. In *ACL*, 1722–1731.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. *ICLR*. Accepted as poster.

Wang, D.; Zhang, X.; Yu, D.; Xu, G.; and Deng, S. 2020. CAME: Content-and Context-Aware Music Embedding for Recommendation. *IEEE Transactions on Neural Networks and Learning Systems*.

Wang, P.; Xu, B.; Wu, Y.; and Zhou, X. 2015. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1): 1–38.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019. Heterogeneous Graph Attention Network. The WebConf '19, 2022–2032. New York, NY, USA: ACM. ISBN 978-1-4503-6674-8.

Wishart, D. S.; Knox, C.; Guo, A. C.; Cheng, D.; Shrivastava, S.; Tzur, D.; Gautam, B.; and Hassanali, M. 2008. DrugBank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic acids research*, 36(suppl_1): D901–D906.

Wu, L.; Quan, C.; Li, C.; Wang, Q.; Zheng, B.; and Luo, X. 2019. A context-aware user-item representation learning for item recommendation. *ACM Transactions on Information Systems (TOIS)*, 37(2): 1–29.

Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; and Chang, E. Y. 2015. Network Representation Learning with Rich Text Information. In *IJCAI*.

Yu, D.; Wanyan, W.; and Wang, D. 2021. Leveraging contextual influence and user preferences for point-of-interest recommendation. *Multimedia Tools and Applications*, 80(1): 1487–1501.

Yujun, C.; Yuanhong, W.; Yutao, Z.; Juhua, P.; and Xiangliang, Z. 2019. AMENDER: an Attentive and Aggregate Multi-layered Network for Dataset Recommendation. In *ICDM*.

Zhang, Z.; Li, X.; and Gan, C. 2021. Identifying influential nodes in social networks via community structure and influence distribution difference. *Digital Communications and Networks*, 7(1): 131–139.

Zheng, J.; Liu, J.; Shi, C.; Zhuang, F.; Li, J.; and Wu, B. 2016. Dual similarity regularization for recommendation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 542–554. Springer.

Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1: 57–81.