GAKE: Graph Aware Knowledge Embedding

Jun Feng[⋆]**, Minlie Huang**[⋆]**, Yang Yang**[†]**, and Xiaoyan Zhu**[⋆]
[⋆] State Key Lab. of Intelligent Technology and Systems, National Lab. for Information Science and Technology
Dept. of Computer Science and Technology, Tsinghua University
[†] College of Computer Science and Technology, Zhejiang University
feng-j13@mails.tsinghua.edu.cn, aihuang@tsinghua.edu.cn
yangya@zju.edu.cn, zxy-dcs@tsinghua.edu.cn

Abstract

Knowledge embedding, which projects triples in a given knowledge base to d-dimensional vectors, has attracted considerable research efforts recently. Most existing approaches treat the given knowledge base as a set of triplets, each of whose representation is then learned separately. However, as a fact, triples are connected and depend on each other. In this paper, we propose a graph aware knowledge embedding method (GAKE), which formulates knowledge base as a directed graph, and learns representations for any vertices or edges by leveraging the graph's structural information. We introduce three types of graph context for embedding: neighbor context, path context, and edge context, each reflects properties of knowledge from different perspectives. We also design an attention mechanism to learn representative power of different vertices or edges. To validate our method, we conduct several experiments on two tasks. Experimental results suggest that our method outperforms several state-of-art knowledge embedding models.

1 Introduction

Knowledge bases, such as DBpedia, YAGO, and Freebase, are important resources to store complex structured facts about the real world in the form of triplets as (*head entity*, *relation*, *tail entity*). These knowledge bases have benefited many applications, such as web search and question answer. In the meanwhile, knowledge base embedding, which aims to learn a D-dimensional vector for each *subject* (i.e., an entity or a relation) in a given knowledge base, has attracted considerable research efforts recently (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015b; Ji et al., 2015). For instance, TransE method (Bordes et al., 2013) regards the relation in a triplet as a translation between the embedding of the two entities. In other words, TransE learns a preference of $h + r = t$ for each triple, where h, r, and t are the representation vector of head entity, relation, and tail entity respectively. Similar ideas are also proposed in TransH (Wang et al., 2014), TransR (Lin et al., 2015b), TransSparse (Ji et al., 2016), etc.

Despite the success of above methods in learning knowledge representations, most of them mainly consider knowledge base as a set of triples and models each triple separately and independently. However, in reality, triples are connected to each other and the whole knowledge base could be regarded as a directed *graph* consisting of vertices (i.e., entities) and directed edges (i.e., relations). In this way, we see that most of existing methods only consider "one hop" information about directed linked entities while miss more global information, such as multiple-steps paths, K-degree neighbors of a given vertex, etc. We call these different structural information as *graph context* inspired by *textural context* utilized in learning a given word's representation (Tomas Mikolov, 2013).

In this paper, we present a novel method to learn the representations of knowledge by utilizing graph context. Figure 1 gives an example to further explain the motivation of our work. In Figure 1(a), we are given a knowledge base organized as a directed graph which shores the facts about the singer Taylor Swift and president Barack Obama. We then demonstrate three kinds of graph context utilized to encode "Taylor_Swift" and "Barack_Obama".

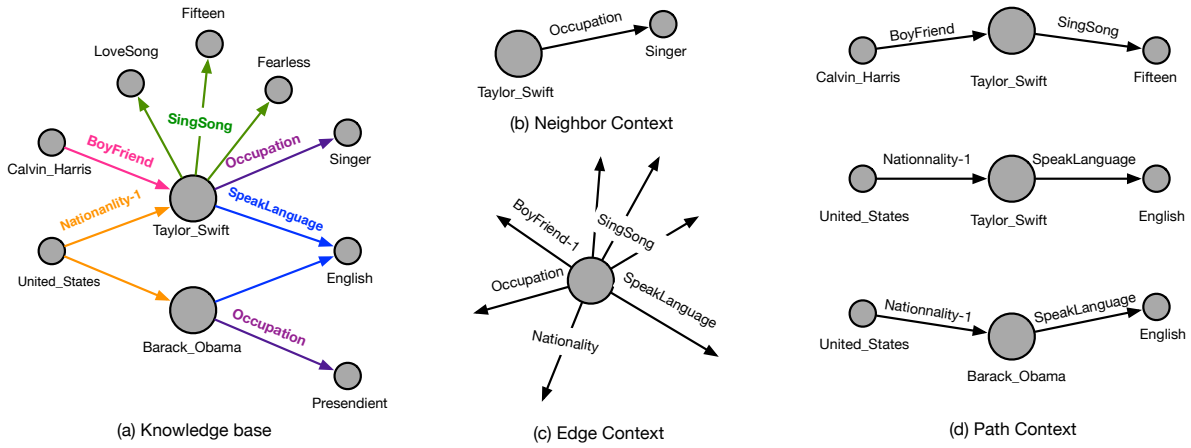

Figure 1: An illustration of three types of graph context, given by a knowledge base.

Neighbor context, as shown in Figure 1(b), consists of the target entity (e.g., "Taylor_Swift") and its directed linked entities (e.g., "Singer") along with their relations (e.g., "Occupation"). It is the most common context and is used in all knowledge base embedding methods.

Edge context, which is shown in Figure 1(c), indicates all kinds of relations relevant to the target entity, such as "SingSong", "BoyFriend^{-1}"(a reverse relation of 'BoyFriend"), "Nationality", and "Occupation" relations of "Taylor_Swift". The relations together would be helpful identify the target entity. For example, "SingSong" and several "BoyFriend^{-1}" relations represent the fact that Taylor, as a singer, has quite a few boy friends in reality. Please notice that different relations has different representation power. For instance, "SingSong" is a very unique relation and is very helpful to identify a singer. Meanwhile, "Nationality" occurs with every human being, so that gains less value. We will introduce how to handle this issue by utilizing an *attention mechanism* in our proposed method latter.

Path context is defined as paths in the given graph containing the target entity. Figure 1(d) gives an example of several 3-step paths containing "Taylor_Swift" or "Barack_Obama". The two paths $United\_States \xrightarrow{Nationality^{-1}} Taylor\_Swift/Barack\_Obama \xrightarrow{SpeakLanguage} English$ represent that the two target entities are similar in terms of nationality and language and suggests their embedding vectors should be somehow similar from this perspective.

There are several challenges when learning knowledge representation by graph context. First, there are quite a few different types of graph context while each has unique structural properties. How to propose a general framework that is able to handle all kinds of graph context is one of the challenges in this work. Second, as we have mentioned previously, in the same type of graph context, different entities/relations have different representation power. For example, in edge context, the "SingSong" relation is more powerful than the "occupation" relation as the former one is less frequent and more unique for singers. How to learn the representation power of each entity/relation is the second challenge we meet. Third, how to estimate model parameters by utilizing real data is also a challenge.

Our contributions in this work include: (1) We treat a given knowledge base as a directed graph instead of a set of independent triples, and extract different types of graph context to study the representation of knowledge. (2) We propose a novel and general representation learning approach, GAKE (Graph Aware Knowledge Embedding), which can be easily extended to consider any type of graph context. (3) We propose an attention mechanism in our approach to learn representation power of different entities and relations.

The rest of this paper are organized as follows. In Section 2, we introduce some related works. In Section 3, we detail the proposed method of graph aware knowledge embedding. Section 4 describes the data and presents experimental results to validate our method. Section 5 concludes the paper.

Table 1: A summary of different knowledge embedding methods.

Method	Triple	Path	Edge
NTN(Socher et al., 2013)	✓	×	×
TransE(Bordes et al., 2013)	✓	×	×
TransH(Wang et al., 2014)	✓	×	×
TransR(Lin et al., 2015b)	✓	×	×
TransD(Ji et al., 2015)	✓	×	×
TranSparse(Ji et al., 2016)	✓	×	×
PTransE(Lin et al., 2015a)	✓	✓	×
Traversing(Gu et al., 2015)	✓	✓	×
GAKE(ours)	✓	✓	✓

2 Related Work

In this section, we review some existing work relevant to our paper. Generally, our work is closely related to the following two topics: (1) knowledge base embedding (2) Graph embedding.

2.1 Knowledge Base Embedding

A variety of approaches have been explored for knowledge base embedding, such as general linear based models, such as SE (Bordes et al., 2011), bilinear based models, like LFM (Jenatton et al., 2012; Sutskever et al., 2009), neural network based models, like SLM (Socher et al., 2013), NTN (Socher et al., 2013), and translation based models (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015a). The mainstream models for knowledge base embedding are translation based models including TransE (Bordes et al., 2013) and its variant models.

Translation-based models all share quite similar principle $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, where $\mathbf{h}, \mathbf{r}$ and $\mathbf{t}$ are the embedding vectors of a triple (h, r, t), though these models differ in score functions. The score function of the translation based models is designed as: $f_r(h, t) = \mathbf{h}_r + \mathbf{r} - \mathbf{t}_r$, where $\mathbf{h}_r$ and $\mathbf{t}_r$ are the embedding vectors of head and tail entities which projected into the relation-specific space.

In TransE (Bordes et al., 2013), the entity and relation embedding vectors are in the same space, say $\mathbf{h}_r = \mathbf{h}, \mathbf{t}_r = \mathbf{t}$. In TransH (Wang et al., 2014), entity embedding vectors are projected into a relation-specific hyperplane $\mathbf{w}_r$, say $\mathbf{h}_r = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h}\mathbf{w}_r, \mathbf{t}_r = \mathbf{t} - \mathbf{w}_r^\top \mathbf{t}\mathbf{w}_r$. In TransR (Lin et al., 2015b), $\mathbf{h}_r = \mathbf{h}\mathbf{M}_r, \mathbf{t}_r = \mathbf{t}\mathbf{M}_r$, where entities are projected from the entity space to the relation space by $\mathbf{M}_r$. In TransD (Ji et al., 2015), $\mathbf{h}_r = \mathbf{M}_{rh}\mathbf{h}$, $\mathbf{t}_r = \mathbf{M}_{rt}t$, where the mapping matrices $\mathbf{M}_{rh}$ and $\mathbf{M}_{rt}$ are both related to the entity and relation. In TransSparse (Ji et al., 2016), $\mathbf{h}_r = \mathbf{M}_r(\theta_r)\mathbf{h}$, $\mathbf{t}_r = \mathbf{M}_r(\theta_r)\mathbf{t}$, where $\mathbf{M}_r$ is an adaptive sparse matrix, whose sparse degrees are determined by the number of entities linked by the relations.

In addition, there are still some works(Xiao et al., 2016b; Xiao et al., 2016a) follow the principle $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, although they do not share the same form of score function. Particularly, (Xiao et al., 2016b) proposes to use a generative model to deal with multiple semantic meanings of a relation. To accommondate more flexible knowledge embedding, (Xiao et al., 2016a) proposes a manifold principle instead of a point-wise estimation of entity and relation embeddings. There are some other works incorporate additional information, such as text(Toutanova and Chen, 2015; Toutanova et al., 2015) and entity types(Guo et al., 2015).

Above knowledge base embedding models all treat the knowledge base as a set of triples. However, in fact, knowledge base is a graph with its graph structure which can be used to better embed the entities and relations in knowledge base. Although (Gu et al., 2015) and PTransE(Lin et al., 2015a) introduce the relation path instead of only considering the direct relations between entities, they just treat the relation path as a new relation and the path length is limited to the model complexity.

However, (Feng et al., 2016) claims the principle $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ is too strict to model the complex and diverse entities and relations and propose a novel principle Flexible Translation to address these issues without increasing the model complexity.

Table 1 compares different knowledge representation learning methods by types of information each method considers. In the table, *Triple* means using each fact as the context when embedding an entity

(or a relation); *Path* stands for treating multiple steps of (undirected) linked entities as the context; *Edge* indicates using all relations that connect to the target entity as the context.

2.2 Graph Embedding

A growing literature has been studying the embedding of graph structure. For example, DeepWalk (Perozzi et al., 2014) uses local information obtained from truncated random walks to learn latent representations by treating walks as the equivalent of sentences. Line (Tang et al., 2015) is a network embedding method that preserves both the local and global network structures.

Although the graph embedding models use the network structures to learn the latent representations, the proposed models are still not suit for us to learn the embeddings of knowledge base. The first reason is that, the knowledge base embedding should learn the representations of both entities(vertices) and relations(edges), but network embedding models only learn the representations for vertices. Second, the assumptions which is the foundation of their models do not hold in knowledge base. For instance, in Line (Tang et al., 2015), it assumes that two vertices which are connected through a strong tie should be similar and be placed closely. But, in knowledge base the head entity and tail entity of a triple may be totally different, such as in triple *(Barack_Obama, Gender,Male)*, entity "Barack_Obama" and "Male" are not the same at all.

In this paper, we propose a novel approach to learn the representations of entities and relations by formulating a given knowledge base as a directed graph and leveraging the graph's structural information.

3 Our Approach

In the following, we present our approach, GAKE (Graph Aware Knowledge Embedding), for learning representations of a given knowledge graph. We describe our approach in steps, adding complexity, and start with necessary notations and definitions.

3.1 Preliminaries

A traditional knowledge graph is a set of triples, each describes a fact, as (*Barack_Obama*, *SpeakLanguage*, *English*). In this work, we use a directed graph to represent these facts by treating head/tail entities as vertices and relations as directed edges. More formally, we have

Definition 1 (Knowledge Graph) *A knowledge graph $G = (V, E)$ is a directed graph, where V is the set of vertices (i.e., entities), and E is the set of edges, where each directed edge $e = (v_i, v_j)$ represents the relation from the entity v_i to the entity v_j $(v_i, v_j \in V)$.*

The way to build a knowledge graph as we defined from given facts (or triples) is as follows: for each fact (h, t, r), where h and t are two terms to represent head entity and tail entity respectively, we first create two corresponding vertices v_i and v_j in the graph G, where i and j are unique index of h and t respectively. After that, we create a directed edge e, which represents the relation r, from v_i to v_j, along with a reverse relation r^{-1} from v_j to v_i. This is a common trick, which is similar to "back translation" in machine translation, to allow us to fully utilize the structural information of knowledge graph and improve the performance. The above process keeps running until all facts are included in the graph G.

Moreover, we use $s = (t, k)$ to represent a *subject* (i.e., a vertex or an edge) of the knowledge graph G, where t indicates subject type, and k is the index of the corresponding vertex or edge. Specifically, we let $t = 0$ to denote a vertex and let $t = 1$ to denote an edge. We use a set $S = \{s_i\}$ to represent all subjects in G.

Given a subject s_i, we define its *context* as a set of other subjects to indicates vertices or edges relevant to s_i:

Definition 2 (Graph Context) *Given a subject s_i, its graph context $c(s_i)$ is a set of other subjects relevant to s_i: $\{s_w | s_w \in S, s_w$ relevant to $s_i\}$.*

Different types of graph context defines the "relevance" between subjects differently. In this work, we use three types of graph context as examples, which will be introduced in detail later.

The objective of GAKE is to learn the representation of each subject in a given knowledge graph G according to its graph context. More formally, we target the problem of *Knowledge Graph Embedding* as

Problem 1 (Knowledge Graph Embedding) *Given a knowledge graph $G = (V, E)$, the problem of knowledge graph embedding aims to represent each vertex $v \in V$ and each edge $r \in E$ by a d-dimensional vector with real numbers.*

Then, we introduce te notations used in GAKE. In detail, s is a subject (i.e., a vertex or an edge); $C(s)$ means graph context of the subject s; $\phi(s)$ is embedding vector of the subject s; $\pi(C(s))$ is translation of subject s's context; $a(s)$ means attention model of a given subject s; θ is parameters used in the attention model.

3.2 Framework

We then introduce our approach in detail. Generally, the learning objective of GAKE is to predict missing subjects given by their context. (e.g., given two vertices, predicting whether there is a missing link from one to another). More formally, we define the probability of s_i given one of its contexts $c(s_i)$:

$$P(s_i|c(s_i)) = \frac{exp(\phi(s_i)^\top \pi(c(s_i))}{\sum_{j=1}^{|S|} exp(\phi(s_j)^\top \pi(c(s_i)))} \tag{1}$$

where $\phi : s_i \in S \longmapsto \mathbb{R}^{|S| \times D}$ is the embedding vector of a given subject s_i, and $\pi(\cdot)$ is a function that represents the translation of a graph context. In this work, we define $\pi(\cdot)$ as follows:

$$\pi(c(s_i)) = \frac{1}{|c(s_i)|} \sum_{s_j \in c(s_i)} \phi(s_j) \tag{2}$$

We then introduce how to construct different types of graph context. Specifically, to take advantage of the graph structure, given a subject s_i, we consider three types of context: *neighbor context* $C_N(s_i)$, *path context* $C_P(s_i)$, and *edge context* $C_E(s_i)$. Please notice that we take these context as examples while our approach is flexible and could easily be extended to other types of graph context.

Neighbor context. Given a subject s_i, taking an entity as an example, we regard each of its out-neighbors, along with their relations, as the *neighbor context*. Formally, when s_i is an entity, its neighbor context $c_N(s_i)$ is a pair of subjects (e, v), where v is an another vertex in G and e is a directed edge links s_i and v. In the case of s_i is a relation, its neighbor context $c_N(s_i)$ is a pair (v, v'), where v and v' are two vertices connected by s_i. One thing worth to notice is that neighbor context is equivalent to using triplets relevant to the given subject s_i.

The objective function of taking neighbor context into consideration is to maximize the log-likelihood of all subjects given by their neighbor contexts. Based on Eq. 1, we have

$$O_N = \sum_{s_i \in S} \sum_{c_N(s_i) \in C_N(s_i)} \log p(s_i|c_N(s_i)) \tag{3}$$

where $C_N(s_i)$ is the set of neighbor context of subject s_i.

Path context. A path in a given knowledge graph reflects both direct and indirect relations between entities. For example, the path $v_1 \xrightarrow{BornInCity} v_2 \xrightarrow{CityInState} v_3 \xrightarrow{StateInCountry} v_4$ indicates the relation "Nationality" between v_1 and v_4.

In this work, given a subject s_i, we use *random walk* to collect several paths starting from s_i. For more details, we first sample a integer L uniformly to indicates the length of the path (i.e., number of edges) we aim to generate. After that, at each step, the random walk will choose a neighbor randomly and will terminate once L edges have been collected. We define the path context $c_P(s_i)$ as a set of vertices and edges that are contained in a generated path. Similar methods are also used in (Spielman and Teng, 2004) and (Perozzi et al., 2014).

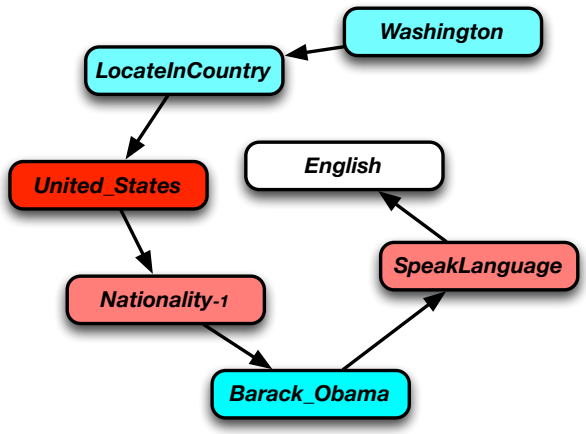

Figure 2: Illustration of the attention for a path context when predicting the entity "English". Darker cells indicate greater attentions.

We then aim to maximize the probability of a subject s_i given by all paths starting from s_i:

$$O_P = \sum_{s_i \in S} \sum_{c_P(s_i) \in C_P(s_i)} \log p(s_i | c_P(s_i)) \tag{4}$$

Edge context. All relations connecting a given entity are representative to that entity, while all entities linked with a given relation are also able to represent that relation. For example, a relation connected with "United Kingdom", "France", "China", and "United States" is most likely to be "Nationality". We define he edge context $c_E(s_i)$ of a subject s_i as all other subjects directly linked with s_i. When s_i is a vertex, $c_E(s_i)$ is a set of edges of s_i, while when s_i is an edge, $c_E(s_i)$ consists of all vertices connected with s_i. Similar with other two types of graph context, we define the objective function of learning knowledge representation when considering edge context as follows:

$$O_E = \sum_{s_i \in S} \log p(s_i | c_E(s_i)) \tag{5}$$

Context extension. To utilize other types of graph context, one could first define $c(s_i)$ and the algorithm used to extract the context from the given knowledge graph G. After that, the remaining steps for knowledge representation learning would be exactly the same with other types of graph context. Thus, our framework is general and flexible to extend different types of graph context easily.

3.3 Attention Mechanism

So far, the translation of a graph context, $\pi(\cdot)$, takes the embedding results of each subject contained in the context equally. However, in reality, different subjects may have different power of influence to represent the target subject. As an example shown in Figure 1, in edge context, "SingSong" relation is more unique and preventative than "Nationality" as only few people like singers will connect with this "SingSong" while everyone has "Nationality". In this work, we model representative powers of different subjects in graph context by an *attention mechanism* (Ling et al., 2015; Hermann et al., 2015).

The basic idea of the attention mechanism is using an *attention model* $a(s_i)$ to represent how subject s_i selectively focuses on representing another subject s_j when s_i is a part of s_j's context (Kelvin Xu, 2015). In this work, we define the attention model $a(s_i)$ as

$$a(s_i) = \frac{\exp(\theta_i)}{\sum_{s_j \in C(s_i)} \exp(\theta_j)} \tag{6}$$

where θ is the parameters we aim to estimate. Figure 2 illustrates the attention for a path context when predicting the entity "English", where darker color indicates a greater attention. We see that entities like

"Washington" and relations like "LocateInCountry" have less attentions, while the entity "UnitedStates" and the relation "SpeakLanguage" have greater attentions on representing "English".

We then re-define the translation of a given graph context, taking the embedding vector of each subject with different weights by further considering attention mechanism. Specifically, we have

$$\pi(c(s_i)) = \sum_{s_j \in c(s_i)} a(s_j)\phi(s_j) \tag{7}$$

3.4 Model Learning

To utilize these three types of context, we combine them by jointly maximizing the objective functions:

$$O = \lambda_N O_N + \lambda_P O_P + \lambda_E O_E \tag{8}$$

We define λ_T, λ_P and λ_N to represent the prestige of neighbor context, path context and edge context separately. We then use a Stochastic gradient descent (SGD) algorithm to estimate model parameters by optimizing Eq. 8. The derivatives are calculated using the back-propagation algorithm. The learning rate for SGD is initially set to 0.1 at first and decreased linearly with the number of training instances. Furthermore, to speed up the training process, we use Hierarchical Softmax (Bengio et al., 2006; Mikolov et al., 2013) to reduce the time complexity of normalization.

4 Experiments

We evaluate our proposed approach with two experiments: (1) triple classification (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015b), which determines whether a given triple is correct or not, and (2) link prediction (Wang et al., 2014; Xiao et al., 2016b), which aims to predict missing entities. For the data, we adopt dataset from Freebase (Bollacker et al., 2008): FB15K (Bordes et al., 2013). We then demonstrate the effectiveness of GAKE in the two tasks respectively. In all experiments, we set the dimension of embedding vectors to 100, $\lambda_T = 1$, , $\lambda_P = 0.1$ and $\lambda_E = 0.1$. The code and data used in this work are publicly available[1].

4.1 Triple classification.

Setup. In this task, given a knowledge base and a triple (h, r, t), we aim to determine whether it is correct (i.e., existing in the given knowledge base) or not. This task is also constructed in several previous work (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015b) and is widely used in many NLP scenarios such as question answering. For example, the result of triple classification can be directly applied to answer questions like "Does Taylor Swift publish the song Fifteen". We use the data set FB15K (Lin et al., 2015b), which contains 1,345 relations among 14,951 entities. We use 483,142 triples as training data to learn embeddings of all subjects. We then use 50,000 triples as validation data and 59,071 triples as test data.

We compare the proposed GAKE method with several state-of-art knowledge base embedding baselines, which includes NTN (Socher et al., 2013), TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransR (Lin et al., 2015b) and TransD (Ji et al., 2015). For each baseline method, we first learn representations of all entities and relations. For a query (h, r, t), we define a relation-specific threshold ρ_r by maximizing the classification accuracy on validation set. After that, we calculate the conditional probability $P(t|h, r)$ by regarding h and r as the context of t, while in GAKE, we construct the neighbor context with h and r's corresponding subjects. At last, we say (h, r, t) is positive (correct) if $P(t|h, r) \geq \rho_r$, where ρ_r is estimated according to the validation data.

Results. We show the evaluation results on triple classification in Figure 3. As the figure shows, it is clear that our approach outperforms others by 11.04% in terms of accuracy on average, as the graph context brings more information especially indirect relations between entities when learning the knowledge representations.

[1] https://github.com/JuneFeng/GAKE

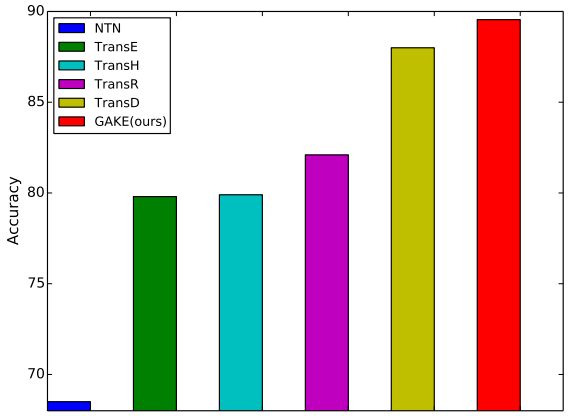

Figure 3: Evaluation results of triple classification.

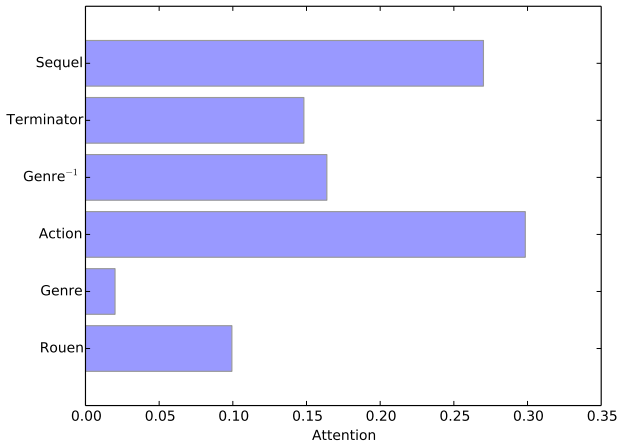

Figure 4: Attentions of subjects as the path context of the entity "Terminate2:JudgementDay".

Furthermore, to better understand the attention mechanism in our approach, we demonstrate attentions of 6 different subjects when they are regarded as the path context of the entity "Terminate2:JudgementDay", which indicates a movie. Figure 4 shows the results. From the figure, we see that two entities, "Action" and "Sequel", have the largest attention to represent the target entity, as "Action" reflects the type of the movie while only some of the movies have sequels. Meanwhile, the relation "Genre" has the least attention as every movie entity connects with "Genre".

4.2 Link Prediction.

Setup. As reported in (Bordes et al., 2011; Bordes et al., 2013), link prediction is to predict the missing h or t given (h, r) or (r, t) respectively. In this task, we conduct the evaluation by ranking the set of candidate entities in knowledge graph, instead of offering a best matching entity. This experiment is conducted on FB15K.

For the baseline methods, we compare our model models with the baselines which include Unstructured (Bordes et al., 2014), RESCAl (Nickel et al., 2011), SE (Bordes et al., 2011), SME(linear/bilinear) (Bordes et al., 2014), LFM (Jenatton et al., 2012) and TransE (Bordes et al., 2013).

Following the protocol in TransE (Bordes et al., 2013), for each test triple (h, r, t), we replace the head entity h by every entity in the knowledge graph, and rank these corrupted triples in descending order by the similarity score which is given by f_r. Similarly, we repeat this procedure by replacing the

Table 2: Experimental results on link prediction.

Data Sets	FB15K			
Metric	Mean Rank		Hits@10(%)	
	Raw	Filter	Raw	Filter
Unstructured (Bordes et al., 2014)	1,074	979	4.5	6.3
RESCAl (Nickel et al., 2011)	828	683	28.4	44.1
SE (Bordes et al., 2011)	273	162	28.8	39.8
SME (linear) (Bordes et al., 2014)	274	154	30.7	40.8
SME (bilinear) (Bordes et al., 2014)	284	158	31.3	41.3
LFM (Jenatton et al., 2012)	283	164	26.0	33.1
TransE (Bordes et al., 2013)	243	125	34.9	47.1
GAKE (ours)	**228**	**119**	**44.5**	**64.8**

tail entity t. After collecting all these triples, we use two evaluation metrics: the mean rank of the correct entities (denotes as *Mean Rank*); the proportion of correct entity ranks within 10 (denotes as *Hits@10*). We expect lower *Mean Rank* and higher *Hits@10* for a better predictor. However, some corrupted triples should be considered as correct ones, since they actually exist in knowledge graph. Ranking such triples ahead of the original correct one should not be counted as an error. To eliminate such cases, we filter out those corrupted triples which appear either in the training, validation or test datasets. We term the former evaluation setting as "Raw" and the latter as "Filter".

Results. Table 2 lists the results on link prediction. It shows that our method GAKE, gets better experiment results than other baselines including Untructured, RESCAL, SE, SME (linear/bilinear), LFM and TransE models. The result demonstrates the superiority of the idea that fully utilizes the graph information to learn representations for entities and relations.

5 Conclusion

In this paper, we propose a graph aware knowledge embedding model to address graph-level contexts. Most existing methods regard knowledge graph as a set of independent triples, and ignore the indirect dependency between subjects (i.e., entities or relations). To deal with this issue, we propose a novel method, GAKE, for learning the representation of a given knowledge graph by formulating a given knowledge base as a directed graph and leveraging graph context, which includes path context, neighbor context, and edge context. We further design an attention mechanism to learn representative power of different subjects. To validate our model, we conduct extensive experiments on benchmark datasets for two tasks, i.e., triple classification and link classification. Experimental results show that GAKE outperforms several state-of-art knowledge embedding methods.

Learning knowledge graph representations is an interesting and new research direction, and there are many potential future directions for this work. For instance, it will be interesting to incorporate the power of explicit knowledge (Wang et al., 2015) into our method to further improve the performance. In addition, the framework of this model is flexible to handle sundry information except the graph context. In other words, we can also build text context by using descriptions of entities or additional text information from other sources like Wikipedia.

6 Acknowledgements

This work was partly supported by the National Basic Research Program (973 Program) under grant No. 2013CB329403, the National Science Foundation of China under grant No.61272227/61332007.

References

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. 2011. Learning structured embeddings of knowledge bases. In *AAAI*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259.

Jun Feng, Minlie Huang, Mingdong Wang, Mantong Zhou, Yu Hao, and Xiaoyan Zhu. 2016. Knowledge graph embedding by flexible translation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016.*, pages 557–560.

Kelvin Gu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. *EMNLP*.

Shu Guo, Quan Wang, Bin Wang, Lihong Wang, and Li Guo. 2015. Semantically smooth knowledge graph embedding. In *Proceedings of ACL*, pages 84–94.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*, pages 3167–3175.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of ACL*, pages 687–696.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *2016 AAAI Spring Symposium Series*.

Ryan Kiros Kyunghyun Cho Aaron C. Courville Ruslan Salakhutdinov Richard S. Zemel Yoshua Bengio Kelvin Xu, Jimmy Ba. 2015. Show, attend and tell: Neural image caption generation with visual attention. *ICML'15*.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2015a. Modeling relation paths for representation learning of knowledge bases.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015b. Learning entity and relation embeddings for knowledge graph completion.

Wang Ling, Lin Chu-Cheng, Yulia Tsvetkov, and Silvio Amir. 2015. Not all contexts are created equal: Better word representations with variable attention. In *ACL*. Citeseer.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816.

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

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

Daniel A Spielman and Shang-Hua Teng. 2004. Nearly-linear time algorithms for graph partitioning, graph ification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90. ACM.

Ilya Sutskever, Joshua B Tenenbaum, and Ruslan Salakhutdinov. 2009. Modelling relational data using bayesian clustered tensor factorization. In *Advances in neural information processing systems*, pages 1821–1828.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. International World Wide Web Conferences Steering Committee.

Kai Chen Greg Corrado Jeffrey Dean Tomas Mikolov, Ilya Sutskever. 2013. Distributed representations of words and phrases and their compositionality. *NIPS'13*, pages 3111–3119.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. *ACL Association for Computational Linguistics*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1112–1119.

Chenguang Wang, Yangqiu Song, Ahmed El-Kishky, Dan Roth, Ming Zhang, and Jiawei Han. 2015. Incorporating world knowledge to document clustering via heterogeneous information networks. In *KDD*, pages 1215–1224.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016a. From one point to a manifold: Knowledge graph embedding for precise link prediction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1315–1321.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016b. TransG : A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.