Joint Semantic Relevance Learning with Text Data and Graph Knowledge

Dongxu Zhang[1,3], **Bin Yuan**[1,3], **Dong Wang**[*1,2], **Rong Liu**[1,4]
[1]CSLT, RIIT, Tsinghua University
[2]Tsinghua National Lab for Information Science and Technology
[3]PRIS, Beijing University of Posts and Telecommunications
[4]Huilan Limited, Beijing, P.R. China
{zhangdx,yuanb,lr}@cslt.riit.tsinghua.edu.cn
wangdong99@mails.tsinghua.edu.cn

Abstract

Inferring semantic relevance among entities (e.g., entries of Wikipedia) is important and challenging. According to the information resources, the inference can be categorized into learning with either raw text data, or labeled text data (e.g., wiki page), or graph knowledge (e.g, WordNet). Although graph knowledge tends to be more reliable, text data is much less costly and offers a better coverage.

We show in this paper that different resources are complementary and can be combined to improve semantic learning. Particularly, we present a joint learning approach that learns vectors of entities by leveraging resources of both text data and graph knowledge. The experiments conducted on the semantic relatedness task show that text-based learning works well on general domain tasks, however for tasks in specific domains, joint learning that involves both text data and graph knowledge offers significant improvement.

1 Introduction

With the development of deep learning and the establishment of large knowledge bases, knowledge embedding has gained much interest in natural language processing. In general, knowledge can be represented by some *entities* that represent semantic concepts, plus the *relations* among them. Knowledge embedding involves representing entities of knowledge bases in a low-dimensional continuous space so that the relations among them can be well represented. The embedding can be conducted with different objectives with different tasks in concern. This paper focuses on the semantic learning task which intends to optimize the semantic relevance among entities by knowledge embedding, e.g., inferring appropriate knowledge (entity) vectors.

According to the information resource that is used to learn with, knowledge embedding can be classified into three categories: raw text learning, labeled text learning and graph knowledge learning. In the raw text learning, the entities are treated as words or phrases, and the local word context information in the raw text is used to drive the embedding. Various approaches of word/phrase embedding belong to this category (Huang et al., 2012; Mikolov et al., 2013). In the labeled text learning, the embedding is based on the description text associated to each entity. A simple approach belonging to this category derives the vector of an entity by averaging the word vectors of the description associated to the entity. Essentially, the knowledge used in this learning is the co-occurrence statistics of the words in the descriptions. Finally, in the graph knowledge learning, the relations among entities labeled by people are used to direct the embedding. Representative approaches of this category include TransE (Bordes et al., 2013) and NTN (Socher et al., 2013).

Different information resources possess their respective advantages and disadvantages. Raw text is totally unstructured and unsupervised (no data annotated). The training data is easy to be collected and in most cases, it offers good entity coverage. The shortcoming, however, is that the useful information is often buried in noise and therefore it is not trivial to extract the desired information. Finally, the learning purely relies on word occurrence statistics, which often under-estimates entities that are infrequent in the training data.

labeled text offers a text description for each entity, so it is more supervised than raw text in the sense that some human-specified annotations are involved. However, the supervision is rather weak, since the relations among entities are not explicitly annotated but implicitly encoded within word co-occurrences of entity descriptions. A particular advantage of the labeled text learning is that the entities that are difficult to learn with raw text because of their limited occurrences can be learned by referring to the words in the descriptions, for instance by averaging the vectors of the words.

Finally, graph knowledge is the most structured and supervised information resource. It is annotated by people and therefore is much more clean and reliable, and the relations among entities can be far beyond the ones that are represented by word local contexts as in raw text. Additionally, the learning does not rely on word statistics and so is mostly suitable for new and infrequent entities, for instance those in a specific domain. An obvious disadvantage of graph knowledge is the high cost in data annotation and the low coverage of the entities and relations. The emergence of large-scale public knowledge bases such as Freebase and Yago partly solved the problem, however for many infrequent entities, the annotations are far from satisfactory and most of the relations are missing.

Due to the respective advantages and disadvantages of different information resources, it is natural to combine them to provide better knowledge embedding. A number of researches have been conducted in this direction. For example, Yu and Dredze (2014) proposed a method to employ graph knowledge to improve word embedding, and Weston et al. (2013) used text data to assist new relation discovery for graph knowledge bases. Nevertheless, there is not a satisfactory framework to learn with multiple and heterogeneous information resources. Particularly, there is limited investigation on to what extent heterogeneous information can be complementary and how they contribute in different situations.

This paper presents a joint learning approach that learns entity vectors by leveraging resources of both raw and labeled text as well as graph knowledge. We first present a joint text learning approach which learns word and entity vectors together with both raw and labeled text. This is similar to the paragraph vector (PV) model (Le and Mikolov, 2014) though a different training approach is adopted in our study. This joint text learning approach is then combined with the graph knowledge learning to form a joint text and graph learning, by integrating the cost functions of the two learning methods.

The experiments are conducted with three information resources: Wikipedia as the raw and labeled text, WordNet and Yago as the graph knowledge. The entity relatedness task is selected to evaluate the performance of the learning methods. Two scenarios have been conducted, one is based on WordNet and the other is based on Yago. The test on WordNet is a general domain task while the test on Yago is a specific domain task. The experimental results show that the joint text learning offers consistent improvement compared to learning with raw text only. When involving graph knowledge, the performance on the general domain task does not show apparent improvement, however on the specific domain task, a significant performance improvement has been observed. These results confirm the importance of learning with heterogeneous information resources.

The rest of the paper is organized as follows: Section 2 briefly describes the related works, Section 3 presents the joint learning approach. The experiments are presented in Section 4, and some discussions are in Section 5. Section 6 concludes the paper.

2 Related work

Many researches have been conducted to learn semantic relevance from raw text data, labeled text data or graph knowledge bases. Most of the studies learn from single information resource.

For raw text learning, various unsupervised learning algorithms have been proposed to learn word representations from large-scale raw text (Huang et al., 2012; Mikolov et al., 2013; Pennington et al., 2014). These methods hypothesize that statistics of word co-occurrences in contexts involve rich semantic and syntactic information and can be utilized to embed words and/or phrases.

Some approaches have been presented to learn with raw text and labeled text together. Gabrilovich and Markovitch (2007) introduced the explicit semantic analysis which represents a word by its distribution over the labeled wikipedia pages instead of the latent concepts as in LSA (Deerwester et al., 1990) and LDA (Blei et al., 2003). Its great performance owes to learning from the combination of raw text and labeled text (wiki pages labeled by entries) resources. Recently, paragraph vector (PV) was also applied to semantic relevance tasks (Dai et al., 2014), which infers word vectors and paragraph vectors together, as the joint text learning presented in this study.

In the field of relevance learning with graph knowledge, early studies focused on measuring word similarity based on the graph theory, for instance, (Rada et al., 1989; Wu and Palmer, 1994; Resnik, 1995). Recent studies focus on various distributed representation models which embed entities and relations of large knowledge graph databases into a low-dimensional continuous space (Bordes et al., 2013; Socher et al., 2013; Fan et al., 2015).

Various approaches have been proposed to utilize heterogeneous resources. Recently, Riedel et al. (2013) demonstrated that text data can help discovering new relation in graph completing task. Weston et al. (2013) used text data for the same purpose while they used word vectors that may leverage text resources more effectively.

Most recently, joint learning approaches have been proposed to learn from heterogeneous resources. Yu and Dredze (2014) learned word vectors by considering not only the word context in text data but also relations in knowledge bases. Their training algorithm draws close the words that are proximate in both text and the knowledge graph. Xu et al. (2014) considered relation types in the joint training process. Faruqui et al. (2014) learned lexicon knowledge by forcing each word in the lexicon to be close to the corresponding pretrained word vector. These studies demonstrated that learning word vectors with both text data and graph knowledge is beneficial to semantic relevance learning.

This study is an extension to the existing joint learning methods. Particularly, we also learn knowledge embedding from descriptions(labeled text). This is contrary to most of the existing researches which learn the knowledge only from context and knowledge graph. Additionally, a new joint learning framework is presented in this work, which integrates text and graph learning as a unified learning processing. Moreover, the contributions of different resources in different situations will be investigated.

3 Method

This section first presents the joint text learning approach which learns entity vectors based on the descriptions that are extracted from Wikipedia. Then the graph knowledge learning is described. Finally the joint text and graph learning is presented which learns with both text data and graph knowledge.

3.1 Joint text learning

Learning entity vectors with raw text can be simply implemented by treating entities as words (or phrases) and learning them together with other words. There are a number of approaches to learning word vectors (Huang et al., 2012; Mikolov et al., 2013; Pennington et al., 2014). In this study, the skip-gram model implemented in the word2vec tool[1] is adopted.

The simple approach to learning with labeled text is to average the vectors of words involved in the description of the current entity e. This is formulated by:

$$v_e = \frac{1}{|D_e|} \sum_{w \in D_e} v_w$$

where v_e denotes the vector of entity e, and v_w denotes the vector of word w. D_e represents set of word tokens within the description of e, and $|D_e|$ represents the size of D_e.

A better approach is to learn word and entity vectors simultaneously. The training is based on negative sampling (Mikolov et al., 2013), with the cost function defined as follows:

$$\mathcal{L}^{txt} = \sum_{e \in E} \sum_{i=1}^{K} max\{0,\ \gamma_{txt} - v_e^T v_{w_i} + v_e^T v_{w_i'}\}$$

where E is the set of entities to learn, and γ_{txt} is the boundary margin which has been empirically set to 0.5 in this study. (w_i, w_i') is a pair of words for which w_i is sampled from the description of entity e, and w_i' is sampled from a proposal distribution. w_i is constrained to be different each time of sampling for a particular entity. K is the number of samples for each entity. In our study, K is set to 30 unless the description is shorter than 30 words. The proposal distribution for sampling w_i' is set to be the unigram distribution of all the words involved in the descriptions of all the entities. We call this model the joint text learning model. The stochastic gradient descendent (SGD) algorithm is employed to optimize $\mathcal{L}^{txt}$ with respect to the entity and word vectors.

Note that this model is similar to PV-DBOW, a distributed bag-of-words model proposed by Le and Mikolov (2014). In PV-DBOW, paragraphs are represented by paragraph vectors (PV) and are trained together with word vectors. The PVs correspond to the entity vectors in our model. A main difference between our model and the PV-DBOW model is that the cost function of our model is based on the hinge loss, while PV-DBOW uses the softmax function. This new cost function provides almost the same performance but offers a lower computation complexity because we don't need to count the sum of distances of the whole dictionary (which is what softmax does).

[1]`http://code.google.com/p/word2vec`

In the experiments, word vectors that are pre-trained with raw text are used to initialize the entity vectors. This pre-training leads to a big improvement compared with random initialization, as will be shown in Section 4.

3.2 Graph knowledge learning

As mentioned in Section 1, knowledge bases such as WordNet and Yago contain plenty of entities and their relations, leading to complex knowledge graphs. Since these entities and relations are annotated by people, graph knowledge is highly reliable and can be used to embed entities. Note that different knowledge bases contain different types of relations. For WordNet, nearly half of the relations are the hypernym-hyponym (is-a) relation, and for Freebase, the relation types are much more complicated. Although it is possible to learn different relations (Bordes et al., 2013), this study does not consider it since our focus is semantic relatedness instead of relation prediction. More discussions about relation type learning will be given in Section 5.

For this reason, only entity vectors are learned (the global relation vector can be absorbed into the entity vectors). This is similar to the unstructured model in Bordes's early work (Bordes et al., 2013), except that distance between vectors is measured by inner product in our model, while Bordes's work used Euclidean distance. We make this choice for two reasons: firstly, to make the graph knowledge learning consistent with the joint text learning so that their results are comparable, and more importantly they can be combined into a joint text and graph learning as will be presented in the next section; secondly, word vectors trained with raw text can be used to pre-train (initialize) the entity vectors, which has been demonstrated to be highly effective, as will be seen in Section 4.

Similar to the text learning, the negative sampling approach is used to train the model. Denote the related entity pairs defined by the knowledge base by $P = \{P_i; P_i = (e_i^l, e_i^r)\}$. For each pair P_i, the negative sampling corrupts the pair by replacing either the left entity e_i^l or the right entity e_i^r with a randomly selected entity. The learning optimizes the following hinge loss function:

$$\mathcal{L}^{grh} = \sum_{P_i \in P} max\{0, \gamma_{grh} - {v_{e_i^l}}^T v_{e_i^r} + {v_{e'^l_i}}^T v_{e'^r_i}\}$$

where γ_{grh} is the boundary margin which is empirically set to 1.0 in this study, and $({e'}^l_i, {e'}^r_i)$ is the corrupted version of (e_i^l, e_i^r) (only one entity corrupted). Again, the SGD algorithm is used to optimize $\mathcal{L}^{grh}$ with respect to the entity vectors.

3.3 Joint text and graph learning

The joint text learning and the graph knowledge learning can be combined. In fact, the two learning approaches are based on the same measure space (the inner product space) and the objective functions are both hinge loss; additionally, both the learning methods train the model using SGD and negative sampling. This means that they are highly consistent and can be easily combined without much change, except that the objective function is modified to integrate the loss derived from both text and graph knowledge. This is formulated by:

$$\mathcal{L}^{joint} = \mathcal{L}^{txt} + \beta \mathcal{L}^{grh} \qquad (1)$$

where β is a hyper-parameter that is set to balance the contributions of the text data and the graph knowledge. The SGD algorithm is employed to optimize $\mathcal{L}^{joint}$ with respect to the entity vectors and the vectors of words that are involved in the entity descriptions. In practice, an iterative strategy is adopted in this work, which performs the joint text learning and the graph knowledge training alternatively and iteratively, with their respective negative sampling schemes applied.

4 Experiments

This section reports the experimental settings and results. The semantic relatedness task was chosen in the study, which measures semantic relatedness among entities and compare the measurements with human-specified scores. We start by presenting the databases, and then report the results on a general domain task and a specific domain task. The data sets and codes are available online. [2]

4.1 Databases

Training data

Three information resources are used in the experiments: Wikipedia as the raw and labeled text, Wordnet and Yago as the graph knowledge. Wikipedia[3] is a free-access, free content internet encyclopedia which at present contains more than 4.7 millions of English entries. Wikipedia itself offers plenty of information, including the main

[2] `http://git.cslt.org/zhangdx/jointsemanticlearning`

[3] `http://wikipedia.org`

content in plain text (description), category information, links to other entries, and info-boxes. Only the plain text and the entry name (title) of descriptions are used in this study. WordNet (Fellbaum, 1998) is a well-known semantic knowledge base which contains 117k English words and the associated information such as brief text descriptions and relations. Yago (Fabian et al., 2007) is another popular semantic knowledge base, derived from Wikipedia, WordNet and GeoNames.

The training (entity vector embedding) is conducted on two development data sets: a subset of entities of WordNet that involves only nouns (WNet-N) and a subset of entities of Yago that involves animal names (Yago-A). WNet-N can be regarded as a data set in the general domain, while Yago-A is a data set in a particular domain. For each entity in the two data sets, the corresponding wiki page is retrieved from Wikipedia, from which the plain text is retrieved and used as the labeled text for the entity. The plain text of all the entities in WNet-N and Yago-A are used as raw text. More details of the data sets are described as follows.

- WNet-N: A subset of WordNet which contains 68569 entities and 70040 relation pairs. All the entities are nouns and are words or phrases in the general domain. 36519 entities find their text descriptions in Wikipedia (labeled text), resulting in 111MB raw text.

- Yago-A: A subtree of Yago which contains 39900 entities, 72936 relation pairs. All the entities are Animal names, which means the entities are domain-specific. 6415 entities find their text descriptions (labeled text), resulting in 19MB raw text.

Note that both WNet-N and Yago-A maintain a connected graph structure which means that for any two entities in the graph, there is at least one path that connects them. This enables the simple connection-based relevance inference which derives relateness of two entities as the connection strength between them in the knowledge graph. Section 5 will compare this simple approach with our proposal.

Test data

As mentioned, this study chooses the semantic relatedness task to evaluate the performance of learned entity vectors. This task computes the relevance (distance) of two entities and then compares the resulting relevance score with the human-specified score. The Spearman coefficient is a widely-adopted metric to evaluate correlation between two variables and is used in this study to measure the consistence of the derived relevance with the human specification.

To test the performance on WNet-N, a subset of WordSimilarity-353[4] is used. The WordSimilarity-353 collection is a well-known test set for semantic relatedness tasks, which contains 353 word pairs and the relatedness scores of all the pairs are manually annotated. After filtering out the words that are absent from the entities of WNet-N, the resulted 301 pairs are used as the test set, named as Sim-301 in the following sections.

For the test on Yago-A, we propose a new test set Animal-143, which contains 143 pairs of common animal names and 92 different animals including mammals, birds, insects and marine animals. All the names are entities of Yago-A. The relatedness score of each pair has been evaluated by 9 persons, and the average is used as the human judgement. The range of score is from 0 to 3. For instance, the score between antelope and swan should probably be 0, and the score between cattle and bison can be 3. Table 1 summarizes the data sets used in the experiments.

4.2 Individual learning

The first experiment studies the performance of learning with raw text, labeled text and graph knowledge individually. As mentioned already, the test are conducted on two data sets: WNet-N and Yago-A, which represent a general domain task and a specific domain task, respectively. The impact of the dimension of the entity vectors is also investigated. The results in terms of Spearman coefficients are reported in Table 2.

For the raw text learning, the entities are treated as words or word sequences (phrases). The vectors of these words and word sequences are then learned by the word2vector tool, together with other words. The raw text data of WNet-N and Yago-A are merged, and combined with additional 200MB plain text to form a training data set to conduct the word vector training. Using the text of both the two data sets is to demonstrate the advantage that word vectors can be learned with out-of-domain data. Note that multi-word entities can be learned as phrase vectors (Mikolov et al., 2013), though this is not considered in this study since the two test sets Sim-301 and Animal-143 contain only single-word entities. The results with

[4]`http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/`

Training Set	Graph Knowledge	labeled Text	Raw Text	Test Set
WNet-N	68569 entities 70040 relations	36519 entities	111MB	Sim-301
Yago-A	39900 entities 72936 relations	6415 entites	19MB	Animal-143

Table 1: Data sets for knowledge embedding and relatedness test.

raw text learning are reported in the row denoted by 'Word2vec' in Table 2.

For the labeled text learning, the entity vectors are derived as the mean vectors of the words involved in the text descriptions. This approach in fact involves both raw text learning (for word vectors) and labeled text learning (vector average), however the main knowledge source is the entity labels of the descriptions. The results with labeled text learning are reported in the row denoted by 'LT-mean' in Table 2.

Different from the mean-vector approach which first trains word vectors and then derives entity vectors, the joint text learning learns word vectors and entity vectors together. Two configurations are tested: in JT-rand, the entity vectors are randomly initialized, while in JT-prt, the entity vectors are initialized (pre-trained) by corresponding word vectors. Note that all the multi-word entities can not be pre-trained as the phrase vectors are not trained, however this does not much impact the resulting performance since the test sets do not involve multi-word entities.

For the graph knowledge learning, two configurations have been tested as well: with and without pre-training. For those entities that can't be pre-trained, random initialization is employed. The results are reported in the rows denoted by 'GR-rand' and 'GR-prt' in Table 2, for the configurations with and without pre-training, respectively.

From the results, it can be observed that the three learning approaches behave differently on the two test sets. The text-based learning exhibits clear advantage compared to graph knowledge learning on the WNet-N test, however on the Yago-A test, the graph knowledge learning is superior. This can be explained by the fact that WNet-N is in the general and involves popular entities that can be well trained with raw and labeled text, however for Yago-A, most of the entities are domain-specific and so it is not easy to learn the entities (and their relations) from unstructured text data. In this case, the human-specified knowledge, i.e., the relations offered by the graph knowledge, tends to provide the most valuable information. On the other hand, the graph knowledge of the general domain tends to be sparse and noisy, which will be discussed in Sec 5, while the graph knowledge of specific domains are generally less sparse and also quite clean. This also leads to more reliable inference with graph knowledge in specific domains.

Another observation is that the dimension of the entity vectors indeed impacts the performance. A larger dimension tends to perform better, at the cost of higher complexity in model training. In the following experiments, the dimension will be set to 100.

It can be also observed that the mean vector (LT-mean) approach does not work well, probably due to the information loss with the simple average. The joint text learning with pre-training (JT-prt) outperforms both LT-mean and Word2Vec. As JT-prt makes use of both raw text and labeled text, this superiority confirms that learning with heterogeneous information resources is beneficial, and the joint learning (JT-prt) is an appropriate way to utilize heterogeneous information effectively.

Finally, it can be seen that the pre-training with word vectors (trained with raw text) contributes to both the text learning and the graph knowledge learning: the pre-trained systems (JT-prt and GR-prt) significantly outperform the random initialized systems (JT-rand and GR-rand). This from another perspective confirms the importance of involving multiple and heterogeneous information resources in knowledge embedding. In addition, the poor performance of JT-rand is mainly due to the incompleteness and bias of descriptions. And the bad results on GR-rand are probably caused by the loss of relation types in databases and also the loss of the variation of length on edges since every edge is trained equally. Thus, pre-training method helps a lot since additional knowledge can be added in and the incompleteness of both description and graph can be hugely solved.

4.3 Joint text and graph learning

This section reports the experiment with the joint text and graph learning. From the experimental re-

Model	Spearman Coefficient					
	WNet-N			Yago-A		
	50	100	200	50	100	200
Word2vec	0.700	0.720	0.729	0.668	0.681	0.704
LT-mean	0.084	0.091	0.083	0.366	0.421	0.470
JT-rand	0.421	0.447	0.422	0.436	0.466	0.449
JT-prt	**0.726**	**0.744**	**0.749**	0.677	0.704	0.719
GR-rand	0.119	0.178	0.180	0.305	0.303	0.410
GR-prt	0.644	0.690	0.690	**0.726**	**0.727**	**0.718**

Table 2: Experimental results with raw text learning, labeled text learning, joint text learning and graph knowledge learning. Bold numbers shows the highest performance in each column.

sults of the previous section, it has been found that learning with multiple resources is helpful, even if with the simple pre-training. The joint text and graph learning takes into account both word contexts and relations when learning the entity vectors, and so may utilize text data and graph knowledge in a more effective way.

Figure 1 presents the experimental results with the joint text and graph learning. The two curves present the Spearman coefficients on WNet-N and Yago-A respectively. The learning rate of the SGD algorithm is set to 0.01, and the iteration number is set to 200. The dimension of the entity vectors and word vectors is set to 100. According to the cost function (1), the learning is impacted by the hyper-parameter β, so the results with various values of β are reported in Figure 1.

From the results presented in Figure 1, very different patterns on the two test sets are observed: for WNet-N, the best β is close to 0.0, which means that involving graph knowledge in the learning simply reduce the performance. However for Yago-A, the best β is around 1.0, which means that to achieve the best performance, the contribution from text and graph resources should be balanced. This discrepancy on the optimal β can be explained in the same way as in the previous experiment, that Yago-A is a specific domain test so that the entities can not be well trained by either text data or graph knowledge, so the two resources need to be utilized together, which is where the joint training contributes. In practical applications, people should increase the β when domain becomes narrow.

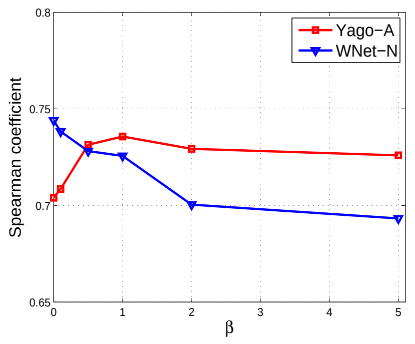

Figure 1: Experimental results with joint text and graph learning, with various values of β.

4.4 Performance comparison

The joint text and graph learning with the individual learning methods are compared in Table 3, where the optimal values of β (0.0 for WNet-N and 1.0 for Yago-A) have been applied. It can be seen that the joint learning contributes significantly to the specific domain task on Yago-A, while for the general domain task on WNet-N, no improvement is found. Nevertheless, since the individual learning is a special case of the joint learning, the latter should be not worse than the former, given that the optimal β is applied.

Model	Spearman Coefficient	
	WNet-N	Yago-A
Word2vec	0.720	0.681
JT-prt	0.744	0.704
GR-prt	0.690	0.727
JTGR-prt	**0.744**	**0.735**

Table 3: Experimental results with joint text and graph learning, where β has been optimized.

5 Discussion

5.1 Performance of graph knowledge learning on different domains

In Section 4, it has been found that graph knowledge training does not work well on the general domain task WNet-N (refer to Table 2). This is possibly caused by the incompleteness of relations when domain becomes wider and the loss of the variation of length on edges in the knowledge base since every edge is trained equally. Notice that, the number of relation pairs in WNet-N is close to Yago-A while entities in WNet-N is more than entities in Yago-A. To further investigate the problem, two simple 'direct inference' algorithms are employed to conduct the tests on WNet-N and Yago-A respectively. The first algorithm is based on the shortest path between two entities in query, and the second one is Wu&Palmer's model (1994) which considers not only the shortest path but also the depth of their common parents. Note that both these two models do not learn any entity vectors but infer relatedness from the relations in the knowledge base, so the results can reflect the quality of the knowledge base.

The results are presented in Table 4. It can be seen clearly that both the shortest-path approach and Wu&Palmer's model show much better performance on Yago-A than on WNet-N. These results provide strong evidence that the general domain is much more complicated, so that the lack of graph knowledge and the problem caused by identical length of edges will easily hurt graph-based inference.

It is also clear that the two direct inference approaches outperform the graph learning approach when no pre-training applied, however with pre-training, the graph learning approach is much more superior, particularly for the WNet-N task. This suggests that learning with broad domain knowledge base is pretty hard, and extra information from raw text is essentially important.

Model	Spearman Coefficient	
	WNet-N	Yago-A
Shortest path	0.312	0.638
Wu&Palmer	0.338	0.662
GR-random	0.178	0.303
GR-prt	0.690	0.727

Table 4: Experimental results with various graph-based entity relateness inference methods.

5.2 Relation type learning

In our experiments, all the relations in the graph knowledge bases are treated indifferently. One may argue that different types of relations should be distinguished and learned distinctively. This is true for some tasks such as relation prediction; however, for the semantic learning task, it is still challenging to use relation information.

Table 5 presents the results with TransE as the graph knowledge learning using all the relation pairs from prolog of WordNet 3.0 which includes 15 types of relations. Note that TransE learns different types of relations as different relation vectors. It can be seen that substituting with TransE has very little effect on performance in both graph knowledge learning (TransE-prt) and joint text and graph learning (JTGR-TransE-prt).

Model	Spearman
JT-prt	0.729
Gr-prt	0.723
TransE-prt	0.726
JTGR-prt	0.739
JTGR-TransE-prt	0.740

Table 5: Performance with TransE in graph knowledge learning.

This can be explained as follows. Intuitively, when people evaluate the relatedness of two entities, both the relation types and the number of relations (directly and indirectly) between them are considered. Although different relation types may impact the judgement differently, learning relation types may force entity vectors to learn to distinguish different types of relations. This is an extra constraint that is irrelevant to our semantic relatedness task. If the constraint is too strong (TransE for example), it may lead to biased learning. Still, relations should be used, but maybe a weak constraint is more appropriate. This is one of the future work.

6 Conclusions

This paper presented a joint text and graph learning method which can learn entity vectors with text data and graph knowledge bases together. We evaluated the proposed method on the semantic relatedness task, and found that involving both text data and graph knowledge does improve performance. Particularly, the experimental results demonstrated that for general domain tasks, the graph knowledge tends to be incomplete thus learning with raw or labeled text is the most effective, however for specific domain tasks, the graph knowledge tends to be more complete, that it can contribute a lot to learning.

Acknowledge

This research was supported by the National Science Foundation of China (NSFC) under the project No. 61371136, and the MESTDC PhD Foundation Project No. 20130002120011. It was also supported by Sinovoice and Huilan Ltd.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022.

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 26*, pages 2787–2795.

Andrew M Dai, Christopher Olah, Quoc V Le, and Greg S Corrado. 2014. Document embedding with paragraph vectors. In *NIPS 2014 in Deep Learning and Representation Learning Workshop*.

Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of The American Society for Information Science*, 41(6):391–407.

MS Fabian, K Gjergji, and W Gerhard. 2007. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *16th International World Wide Web Conference, WWW*, pages 697–706.

Miao Fan, Qiang Zhou, Andrew Abel, Thomas Fang Zheng, and Ralph Grishman. 2015. Probabilistic belief embedding for knowledge base completion. *CoRR*, abs/1505.02433.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, volume 7, pages 1606–1611.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Quoc V Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of The 31st International Conference on Machine Learning*, pages 1188–1196.

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 26*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12.

Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. 1989. Development and application of a metric on semantic nets. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(1):17–30.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *In Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL-HLT*, pages 74–84.

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 26*, pages 926–934.

Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2013)*, pages 1366–1371.

Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics.

Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1219–1228. ACM.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 545–550.