Knowledge Graph and Text Jointly Embedding

Zhen Wang[†§] **, Jianwen Zhang**[†] **, Jianlin Feng**[§] **, Zheng Chen**[†]
[†]{v-zw,jiazhan,zhengc}@microsoft.com
[§]{wangzh56@mail2,fengjlin@mail}.sysu.edu.cn
[†]Microsoft Research [§]Sun Yat-sen University

Abstract

We examine the embedding approach to reason new relational facts from a large-scale knowledge graph and a text corpus. We propose a novel method of jointly embedding entities and words into the same continuous vector space. The embedding process attempts to preserve the relations between entities in the knowledge graph and the concurrences of words in the text corpus. Entity names and Wikipedia anchors are utilized to align the embeddings of entities and words in the same space. Large scale experiments on Freebase and a Wikipedia/NY Times corpus show that jointly embedding brings promising improvement in the accuracy of predicting facts, compared to separately embedding knowledge graphs and text. Particularly, jointly embedding enables the prediction of facts containing entities out of the knowledge graph, which cannot be handled by previous embedding methods. At the same time, concerning the quality of the word embeddings, experiments on the analogical reasoning task show that jointly embedding is comparable to or slightly better than word2vec (Skip-Gram).

1 Introduction

Knowledge graphs such as Freebase (Bollacker et al., 2008) and WordNet (Miller, 1995) have become important resources for many AI & NLP applications such as Q & A. Generally, a knowledge graph is a collection of relational facts that are often represented in the form of a triplet (head entity, relation, tail entity), e.g., "(Obama, Born-in, Honolulu)". An urgent issue for knowledge graphs is the coverage, e.g., even the largest knowledge graph of Freebase is still far from complete.

Recently, targeting knowledge graph completion, a promising paradigm of embedding was proposed, which is able to reason new facts only from the knowledge graph (Bordes et al., 2011; Bordes et al., 2013; Socher et al., 2013; Wang et al., 2014). Generally, in this series of methods, each entity is represented as a k-dimensional vector and each relation is characterized by an operation in $\Re^k$ so that a candidate fact can be asserted by simple vector operations. The embeddings are usually learnt by minimizing a global loss function of all the entities and relations in the knowledge graph. Thus, the vector of an entity may encode global information from the entire graph, and hence scoring a candidate fact by designed vector operations plays a similar role to long range "reasoning" in the graph. However, since this requires the vectors of both entities to score a candidate fact, this type of methods can only complete missing facts for which both entities exist in the knowledge graph. However, a missing fact often contains entities out of the knowledge graph (called *out-of-kb* for short in this paper), e.g., one or both entities are phrases appearing in web text but not included in the knowledge graph yet. How to deal with these facts is a significant obstacle to widely applying the embedding paradigm.

In addition to knowledge embedding, another interesting approach is the word embedding method word2vec (Mikolov et al., 2013b), which shows that learning word embeddings from an unlabeled text corpus can make the vectors connecting the pairs of words of some certain relation almost parallel, e.g., vec("China") $-$ vec("Beijing") $\approx$ vec("Japan") $-$ vec("Tokyo"). However, it does not know the exact relation between the pairs. Thus, it cannot be directly applied to complete knowledge graphs.

The capabilities and limitations of knowledge embedding and word embedding have inspired us to design a mechanism to mosaic the knowledge

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1591–1601,
October 25-29, 2014, Doha, Qatar. ©2014 Association for Computational Linguistics

graph and the "word graph" together in a vector space so that we can score any candidate relational facts between entities and words[1]. Therefore, we propose a novel method to jointly embed entities and words into the same vector space. In our solution, we define a coherent probabilistic model for both knowledge and text, which is composed of three components: the knowledge model, text model, and alignment model. Both the knowledge model and text model use the same core translation assumption for the fact modeling: a candidate fact (h, r, t) is scored based on $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$. The only difference is, in the knowledge model the relation r is explicitly supervised and the goal is to fit the fact triplets, while in the text model we assume any pair of words h and t that concur in some text windows are of certain relation r but r is a hidden variable, and the goal is to fit the concurring pairs of words. The alignment model guarantees the embeddings of entities and words/phrases lie in the same space and impels the two models to enhance each other. Two mechanisms of alignment are introduced in this paper: utilizing names of entities and utilizing Wikipedia anchors. This way of jointly embedding knowledge and text can be considered to be semi-supervised knowledge embedding: the knowledge graph provides explicit supervision of facts while the text corpus provides much more "relation-unlabeled" pairs of words.

We conduct extensive large scale experiments on Freebase and Wikipedia corpus, which show jointly embedding brings promising improvements to the accuracy of predicting facts, compared to separately embedding the knowledge graph and the text corpus, respectively. Particularly, jointly embedding enables the prediction of a candidate fact with out-of-kb entities, which can not be handled by any existing embedding methods. We also use embeddings to provide a prior score to help fact extraction on the benchmark data set of Freebase+NYTimes and also observe very promising improvements. Meanwhile, concerning the quality of word embeddings, experiments on the analogical reasoning task show that jointly embedding is comparable to or slightly better than word2vec (Skip-Gram).

[1]We do not distinguish between "words" and "phrases", i.e., "words" means "words/phrases".

2 Related Work

Knowledge Embedding. A knowledge graph is embedded into a low-dimensional continuous vector space while certain properties of it are preserved (Bordes et al., 2011; Bordes et al., 2013; Socher et al., 2013; Chang et al., 2013; Wang et al., 2014). Generally, each entity is represented as a point in that space while each relation is interpreted as an operation over entity embeddings. For instance, TransE (Bordes et al., 2013) interprets a relation as a *translation* from the head entity to the tail entity. The embedding representations are usually learnt by minimizing a global loss function involving all entities and relations so that each entity embedding encodes both local and global connectivity patterns of the original graph. Thus, we can reason new facts from learnt embeddings.

Word Embedding. Generally, word embeddings are learned from a given text corpus without supervision by predicting the context of each word or predicting the current word given its context (Bengio et al., 2003; Collobert et al., 2011; Mikolov et al., 2013a; Mikolov et al., 2013b). Although relations between words are not explicitly modeled, continuous bag-of-words (CBOW) and Skip-gram (Mikolov et al., 2013a; Mikolov et al., 2013b) learn word embeddings capturing many syntactic and semantic relations between words where a relation is also represented as the *translation* between word embeddings.

Relational Facts Extraction. Another pivotal channel for knowledge graph completion is extracting relational facts from external sources such as free text (Mintz et al., 2009; Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012; Zhang et al., 2013; Fan et al., 2014). This series of methods focuses on identifying local text patterns that express a certain relation and making predictions based on them. However, they have not fully utilized the evidences from a knowledge graph, e.g., knowledge embedding is able to reason new facts without any external sources. Actually, knowledge embedding is very complementary to traditional extraction methods, which was first confirmed by (Weston et al., 2013). To estimate the plausibility of a candidate fact, they added scores from embeddings to scores from an extractor, which showed significant improvement. However, as pointed out in the introduction, their knowledge embedding method cannot predict facts involving out-of-kb entities.

3 Jointly Embedding Knowledge and Text

We will first describe the notation used in this paper. A knowledge graph Δ is a set of triplets in the form $(h, r, t), h, t \in \mathcal{E}$ and $r \in \mathcal{R}$ where $\mathcal{E}$ is the entity vocabulary and $\mathcal{R}$ is a collection of predefined relations. We use bold letters $\mathbf{h}, \mathbf{r}, \mathbf{t}$ to denote the corresponding embedding representations of h, r, t. A text corpus is a sequence of words drawn from the word vocabulary $\mathcal{V}$. Note that we perform some preprocessing to detect phrases in the text and the vocabulary here already includes the phrases. For simplicity's sake, without special explanation, when we say "word(s)", it means "word(s)/phrase(s)". Since we consider triplets involving not only entities but also words, we denote $\mathcal{I} = \mathcal{E} \cup \mathcal{V}$. Additionally, we denote anchors by $\mathcal{A}$.

3.1 Modeling

Our model is composed of three components: the knowledge model, text model, and alignment model.

Before defining the component models, we first define the element model for a fact triplet. Inspired by TransE, we also represent a relation r as a vector $\mathbf{r} \in \Re^k$ and score a fact triplet (h, r, t) by $z(\mathbf{h}, \mathbf{r}, \mathbf{t}) = b - \frac{1}{2}\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|^2$ where b is a constant for bias designated for adjusting the scale for better numerical stability and $b = 7$ is a sensible choice. $z(\mathbf{h}, \mathbf{r}, \mathbf{t})$ is expected to be large if the triplet is true. Based on the same element model of fact, we define the component models as follows.

3.1.1 Knowledge Model

We define the following conditional probability of a fact (h, r, t) in a knowledge graph:

$$\Pr(h|r,t) = \frac{\exp\{z(\mathbf{h}, \mathbf{r}, \mathbf{t})\}}{\sum_{\tilde{h} \in \mathcal{I}} \exp\{z(\tilde{\mathbf{h}}, \mathbf{r}, \mathbf{t})\}} \quad (1)$$

and we have named our model *pTransE* (Probabilistic TransE) to show respect to TransE. We also define $\Pr(r|h,t)$ and $\Pr(t|h,r)$ in the same way by choosing corresponding normalization terms respectively. We define the likelihood of observing a fact triplet as:

$$\begin{aligned}\mathcal{L}_f(h,r,t) = \log \Pr(h|r,t) + \log \Pr(t|h,r) \\ + \log \Pr(r|h,t)\end{aligned} \quad (2)$$

The goal of the knowledge model is to maximize the conditional likelihoods of existing fact triplets in the knowledge graph:

$$\mathcal{L}_K = \sum_{(h,r,t) \in \Delta} \mathcal{L}_f(h,r,t) \quad (3)$$

3.1.2 Text Model

We propose the following key assumption for modeling text, which connects word embedding and knowledge embedding: there are relations between words although we do not know what they are.

Relational Concurrence Assumption. *If two words w and v concur in some context, e.g., a window of text, then there is a relation r_{wv} between the two words. That is, we can state the triplet of (w, r_{wv}, v) is a fact.*

We define the conditional probability $\Pr(w|r_{wv}, v)$ following the same formulation of Eq.(1) to model why two words concur in some context. In contrast to knowledge embedding, here r_{wv} is a hidden variable rather than explicitly supervised.

The challenge is to deal with the hidden variable r_{wv}. Obviously, without any more assumptions, the number of distinct r_{wv} is around $|\mathcal{V}| \times \bar{N}$, where $\bar{N}$ is the average number of unique words concurred with each word. This number is extremely large. Thus it is almost impossible to estimate a vector for each r_{wv}. And the problem is actually ill-posed. We need to constrain the freedom degree of r_{wv}. Here we use auxiliary variables to reduce the size of variables we need to estimate: let $\mathbf{w}' = \mathbf{w} + \mathbf{r}_{wv}$, then

$$z(\mathbf{w}, \mathbf{r}_{wv}, \mathbf{v}) \triangleq z(\mathbf{w}', \mathbf{v}) = b - \frac{1}{2}\|\mathbf{w}' - \mathbf{v}\|^2 \quad (4)$$

and

$$\Pr(w|r_{wv}, v) \triangleq \Pr(w|v) = \frac{\exp\{z(\mathbf{w}', \mathbf{v})\}}{\sum_{\tilde{w} \in \mathcal{V}} \exp\{z(\tilde{\mathbf{w}}', \mathbf{v})\}} \quad (5)$$

In this way we need to estimate vectors $\mathbf{w}$ and $\mathbf{w}'$ for each word w, and a total of $2 \times |\mathcal{V}|$ vectors.

The goal of the text model is to maximize the likelihood of the concurrences of pairs of words in text windows:

$$\mathcal{L}_T = \sum_{(w,v) \in \mathcal{C}} n_{wv} \log \Pr(w|v). \quad (6)$$

In the above equation, $\mathcal{C}$ is all the distinct pairs of words concurring in text windows of a fixed size. And n_{wv} is the number of concurrences of the pair (w, v). Interestingly, as explained in Sec.(3.3), this text model is almost equivalent to Skip-Gram.

3.1.3 Alignment Model

If we only have the knowledge model and text model, the entity embeddings and word embeddings will be in different spaces and any computing between them is meaningless. Thus we need mechanisms to align the two spaces into the same one. We propose two mechanisms in this paper: utilizing Wikipedia anchors, and utilizing names of entities.

Alignment by Wikipedia Anchors. This model is based on the connection between Wikipedia and Freebase: for most Wikipedia (English) pages, there is an unique corresponding entity in Freebase. As a result, for most of the anchors in Wikipedia, each of which refers to a Wikipedia page, we know that the surface phrase v of an anchor actually refers to the Freebase entity e_v. Thus, we define a likelihood for this part of anchors as Eq.(6) but replace the word pair (w, v) with the word-entity pair (w, e_v), i.e., using the corresponding entity e_v rather than the surface word v in Eq.(5):

$$\mathcal{L}_{AA} = \sum_{(w,v)\in\mathcal{C}, v\in\mathcal{A}} \log \Pr(w|e_v) \tag{7}$$

where $\mathcal{A}$ denotes the set of anchors.

In addition to Wikipedia anchors, we can also use an entity linking system with satisfactory performance to produce the pseudo anchors.

Alignment by Names of Entities. Another way is to use the names of entities. For a fact triplet $(h, r, t) \in \Delta$, if h has a name w_h and $w_h \in \mathcal{V}$, then we will generate a new triplet of (w_h, r, t) and add it to the graph. Similarly, we also add (h, r, w_t) and (w_h, r, w_t) into the graph if the names exist and belong to the word vocabulary. We call this sub-graph containing names the *name graph* and define a likelihood for the name graph by observing its triplets:

$$\mathcal{L}_{AN} = \sum_{(h,r,t)\in\Delta} \mathbf{I}_{[w_h\in\mathcal{V}\wedge w_t\in\mathcal{V}]}\cdot\mathcal{L}_f(w_h, r, w_t)+ \mathbf{I}_{[w_h\in\mathcal{V}]}\cdot\mathcal{L}_f(w_h, r, t) + \mathbf{I}_{[w_t\in\mathcal{V}]}\cdot\mathcal{L}_f(h, r, w_t) \tag{8}$$

Both alignment models have advantages and disadvantages. Alignment by names of entities is straightforward and does not rely on additional data sources. The number of triplets generated by the names is also large and can significantly change the results. However, this model is risky. On the one hand, the name of an entity is ambiguous because different entities sometimes have the same name so that the name graph may contaminate the knowledge embedding. On the other hand, an entity often has several different aliases when mentioned in the text but we do not have the complete set, which will break the semantic balance of word embedding. For example, for the entity Apple Inc., suppose we only have the standard name "Apple Inc." but do not have the alias "apple". And for the entity Apple that is fruit, suppose we have the name "apple" included in the name graph. Then the vector of the word "apple" will be biased to the concept of fruit rather than the company. But if no name graph intervenes, the unsupervised word embedding is able to learn a vector that is closer to the concept of the company due to the polarities. Alignment by anchors relies on the additional data source of Wikipedia anchors. Moreover, the number of matched Wikipedia anchors (∼40M) is relatively small compared to the total number of word pairs (∼2.0B in Wikipedia) and hence the contribution is limited. However, the advantage is that the quality of the data is very high and there are no ambiguity/completeness issues.

Considering the above three component models together, the likelihood we maximize is:

$$\mathcal{L} = \mathcal{L}_K + \mathcal{L}_T + \mathcal{L}_A \tag{9}$$

where $\mathcal{L}_A$ could be $\mathcal{L}_{AA}$ or $\mathcal{L}_{AN}$ or $\mathcal{L}_{AA} + \mathcal{L}_{AN}$.

3.2 Training

3.2.1 Approximation to the Normalizers

It is difficult to directly compute the normalizers in $\Pr(h|r, t)$ (or $\Pr(t|h, r)$, $\Pr(r|h, t)$) and $\Pr(w|v)$ as the normalizers sum over $|\mathcal{I}|$ or $|\mathcal{V}|$ terms where both $|\mathcal{I}|$ and $|\mathcal{V}|$ reach tens of millions. To prevent having to exactly calculate the normalizers, we use negative sampling (NEG) (Mikolov et al., 2013b) to transform the original objective, i.e., Eq.(9) to a simple objective of the binary classification problem—differentiating the observed data from noise.

First, we define: (i) the probability of a given triplet (h, r, t) to be true ($D = 1$); and (ii) the probability of a given word pair (w, v) to co-occur ($D = 1$):

$$\Pr(D = 1|h, r, t) = \sigma(z(\mathbf{h}, \mathbf{r}, \mathbf{t})) \tag{10}$$

$$\Pr(D = 1|w, v) = \sigma(z(\mathbf{w}', \mathbf{v})) \tag{11}$$

where $\sigma(x) = \frac{1}{1+\exp\{-x\}}$ and $D \in \{0, 1\}$.

Instead of maximizing $\log \Pr(h|r,t)$ in Eq.(2), we maximize:

$$\log \Pr(1|h,r,t) + \sum_{i=1}^{c} \mathbb{E}_{\tilde{h}_i \sim \Pr_{\text{neg}}(\tilde{h}_i)} [\Pr(0|\tilde{h}_i,r,t)] \quad (12)$$

where c is the number of negative examples to be discriminated for each positive example. NEG guarantees that maximizing Eq.(12) can approximately maximize $\log \Pr(h|r,t)$. Thus, we also replace $\log \Pr(r|h,t)$, $\log \Pr(t|r,h)$ in Eq.(2), and $\log \Pr(w|v)$ in Eq.(6) in the same way by choosing corresponding negative distributions respectively. As a result, the objectives of both the knowledge model $\mathcal{L}_K$ (Eq.(3)) and text model $\mathcal{L}_T$ (Eq.(6)) are free from cumbersome normalizers.

3.2.2 Optimization

We use stochastic gradient descent (SGD) to maximize the simplified objectives.

Knowledge model. Δ is randomly traversed multiple times. When a positive example $(h,r,t) \in \Delta$ is considered, to maximize (12), we construct c negative triplets by sampling elements from an uniform distribution over $\mathcal{I}$ and replacing the head of (h,r,t). The transformed objective of $\log \Pr(r|h,t)$ is maximized in the same manner, but by sampling from a uniform distribution over $\mathcal{R}$ and corrupting the relation of (h,r,t). After a mini-batch, computed gradients are used to update the involved embeddings.

Text model. The text corpus is traversed one or more times. When current word v and a context word w are considered, c words are sampled from the unigram distribution raised to the 3/4rd power and regarded as negative examples $(\tilde{w}, v)$ that are never concurrent. Then we compute and update the related gradients.

Alignment model. $\mathcal{L}_{AA}$ and $\mathcal{L}_{AN}$ are absorbed by the text model and knowledge model respectively, since anchors are considered to predict context given an entity and the name graph are homogeneous to the original knowledge graph.

Joint. All three component objectives are *simultaneously* optimized. To deal with large-scale data, we implement a multi-thread version with shared memory. Each thread is in charge of a portion of the data (either knowledge or text corpus), and traverses through them, calculates gradients and commits the update to the global model and is stored in a block of shared memory. For the sake of efficiency, no lock is used on the shared memory.

Table 1: **Data**: triplets used in our experiments.

#$\mathcal{R}$	#$\mathcal{E}$	#Triplet (Train/Valid/Test)		
4,490	43,793,608	123,062,855	40,528,963	40,528,963

3.3 Connections to Related Models

TransE. (Bordes et al., 2013) proposed to model a relation r as a translation vector $\mathbf{r} \in \Re^k$ which is expected to connect $\mathbf{h}$ and $\mathbf{t}$ with low error if $(h,r,t) \in \Delta$. We also follow it. However, TransE uses a margin based ranking loss $\{\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|^2+\gamma-\|\tilde{\mathbf{h}}+\mathbf{r}-\tilde{\mathbf{t}}\|^2\}_+$. It is not a probabilistic model and hence it needs to restrict the norm of either entity embedding and/or relation embedding. Bordes et al. (2013) intuitively addresses this problem by simply normalizing the entity embeddings to the unit sphere before computing gradients at each iteration. We define pTransE as a probabilistic model, which doesn't need additional constraints on the norms of embeddings of entities/words/relations, and thus eliminates the normalization operations.

Skip-gram. (Mikolov et al., 2013a; Mikolov et al., 2013b) defines the probability of the concurrence of two words in a window as:

$$\Pr(w|v) = \frac{\exp\{\mathbf{w}'^{\mathrm{T}}\mathbf{v}\}}{\sum_{\tilde{w}\in\mathcal{V}} \exp\{\tilde{\mathbf{w}}'^{\mathrm{T}}\mathbf{v}\}} \quad (13)$$

which is based on the inner product, while our text model (Eqs. (4), (5)) is based on distance. If we constrain $\|\mathbf{w}\| = 1$ for each w, then $\mathbf{w}'^{\mathrm{T}}\mathbf{v} = 1 - \frac{1}{2}\|\mathbf{w}' - \mathbf{v}\|^2$. It is easy to see that our text model is equivalent to Skip-gram in this case. Our distance-based text model is directly derived from the triplet fact model, which clearly explains why it is able to make the pairs of entities of a certain relation parallel in the vector space.

4 Experiments

We empirically evaluate and compare related models with regards to three tasks: triplet classification (Socher et al., 2013), improving relation extraction (Weston et al., 2013), and the analogical reasoning task (Mikolov et al., 2013a). The related models include: for knowledge embedding alone, TransE (Bordes et al., 2013), pTransE (proposed in this paper); for word embedding alone, Skip-gram (Mikolov et al., 2013b); for both

Table 2: **Data**: the number of $e-e$, $w-e$, $e-w$, $w-w$ triplets/analogies where w represents the out-of-kb entity, which is regarded as word and replaced by its corresponding entity name.

Type	#Triplet (Valid/Test)		#Analogy
$e-e$	12,305,200	12,305,200	71,441
$w-e$	3,655,164	3,654,404	70,878
$e-w$	3,643,914	3,642,978	70,442
$w-w$	460,762	451,381	40,980

knowledge and text, we use "respectively" to refer to the embeddings learnt by TransE/pTransE and Skip-gram, respectively, "jointly" to refer to our jointly embedding method, in which "anchor" and "name" refer to "Alignment by Wikipedia Anchors" and "Alignment by Names of Entities", respectively.

4.1 Data

To learn the embedding representations of entities and words, we use a knowledge graph, a text corpus, and some connections between them.

Knowledge. We adopt Freebase as our knowledge graph. First, we remove the user profiles, version control, and meta data, leaving 52,124,755 entities, 4,490 relations, and 204,120,782 triplets. We call this graph *main facts*. Then we held out 8,331,147 entities from main facts and regard them as out-of-kb entities. Under such a setting, from main facts, we held out all the triplets involving out-of-kb entities, as well as 24,610,400 triplets that don't contain out-of-kb entities. Held-out triplets are used for validation and testing; the remaining triplets are used for training. See Table 1 for the statistics.

We regard out-of-kb entities as words/phrases and thus divide the held-out triplets into four types: no out-of-kb entity ($e-e$), the head is out-of-kb entity but the tail is not ($w-e$), the tail is out-of-kb entity but the head is not ($e-w$), and both the head and tail are out-of-kb entities ($w-w$). Then we replace the out-of-kb entities among the held-out triplets by their corresponding entity names. The mapping from a Freebase entity identifier to its name is done through the Freebase predicate—"/type/object/name". Since some entity names are not present in our vocabulary $\mathcal{V}$, we remove triplets involving these names (see Table 2). In such a way, besides the missing edges between existing entities, the related models can be evaluated on triplets involving words/phrases as their head and/or tail.

Table 3: **Triplet Classification**: comparison between TransE and pTransE over $e-e$ triplets.

Method	Accuracy (%)	Area under PR curve
TransE	93.1	0.86
pTransE	**93.4**	**0.97**

Text. We adopt the Wikipedia (English) corpus. After removing pages designated for navigation, disambiguation, or discussion purposes, there are 3,469,024 articles left. We apply sentence segmentation, tokenization, Part-of-Speech (POS) tagging, and named entity recognition (NER) to these articles using Apache OpenNLP package[2]. Then we conduct some simple chunking to acquire phrases: if several consecutive tokens are identically tagged as "Location"/"Person"/"Organization", or covered by an anchor, we combine them as a chunk. After the preprocessing, our text corpus contain 73,675,188 sentences consisting of 1,522,291,723 chunks. Among them, there are around 20 millions distinct chunks, including words and phrases. We filter out punctuation and rare words/phrases that occur less than three times in the text corpus, reducing $|\mathcal{V}|$ to 5,240,003.

Alignment. One of our alignment models needs Wikipedia anchors. There are around 45 million such anchors in our text corpus and 41,970,548 of them refer to entities in $\mathcal{E}$. Another mechanism utilizes the name graph constructed through names of entities. Specifically, for each training triplet (h,r,t), suppose h and t have entity names w_h and w_t, respectively and $w_h, w_t \in \mathcal{V}$, the training triplet contributes (w_h,r,w_t), (w_h,r,t), and (h,r,w_t) to the name graph. There are 81,753,310 triplets in our name graphs. Note that there is no overlapping between the name graph and held-out triplets of $e-w$, $w-e$, and $w-w$ types.

4.2 Triplet Classification

This task judges whether a triplet (h,r,t) is true or false, i.e., binary classification of a triplet.

Evaluation protocol. Following the same protocol in NTN (Socher et al., 2013), for each true triplet, we construct a false triplet for it by randomly sampling an element from $\mathcal{I}$ to corrupt its head or tail. Since $|\mathcal{E}|$ is significantly larger than $|\mathcal{V}|$ in our data, sampling from a uniform distri-

[2]https://opennlp.apache.org

Table 4: **Triplet classification**: accuracy (%) over various types of triplets.

Type	$e-e$	$w-e$	$e-w$	$w-w$	all
respectively	93.4	52.1	51.4	71.0	77.5
jointly (anchor)	94.4	67.0	66.7	79.8	81.9
jointly (name)	94.5	80.5	80.0	89.0	87.7
jointly (anchor+name)	**95.0**	**82.0**	**81.5**	**90.0**	**88.8**

bution over $\mathcal{I}$ will let triplets involving no word dominate the false triplets. To avoid that, when we corrupt the head of (h, r, t), if $h \in \mathcal{E}$, h' is sampled from $\mathcal{E}$ while if $h \in \mathcal{V}$, h' is sampled from $\mathcal{V}$. The same rule is applied when we corrupt the tail of (h, r, t). In this way, for each of the four types of triplets, we ensure the number of true triplets is equal to that of false ones.

To classify a triplet (h, r, t), we first use the considered methods to score it. TransE scores it by $-|\mathbf{h} + \mathbf{r} - \mathbf{t}|$. Our models score it by $\Pr(D = 1|h, r, t)$ (see Eq.(10)). Then the considered methods label a triplet (h, r, t) as true if its score is larger than the relation-specific threshold of r, as false otherwise. The relation-specific thresholds are chosen to maximize the classification accuracy over the validation set.

We report the classification accuracy. Additionally, we rank all the testing triplets by their scores in descending order. Then we draw a precision-recall (PR) curve based on this ranking and report the area under the PR curve.

Implementation. We implement TransE (Bordes et al., 2013), Skip-gram (Mikolov et al., 2013a), and our models.

First, we train TransE and pTransE over our training triplets with embedding dimension k in $\{50, 100, 150\}$. Adhering to (Bordes et al., 2013), we use the fixed learning rate α in $\{0.005, 0.01, 0.05\}$ for TransE during its 300 epochs. For pTransE, we use the number of negative examples per positive example c among $\{5, 10\}$, the learning rate α among $\{0.01, 0.025\}$ where α decreases along with its 40 epochs. The optimal configurations of TransE are: $k = 100$, $\alpha = 0.01$. The optimal configurations of pTransE are: $k = 100$, $c = 10$, and $\alpha = 0.025$.

Then we train Skip-gram with the embedding dimension k in $\{50, 100, 150\}$, the max skip-range s in $\{5, 10\}$, the number of negative examples per positive example c in $\{5, 10\}$, and learning rate $\alpha = 0.025$ linearly decreasing along with the 6 epochs over our text corpus. Popular words whose frequencies are larger than 10^{-5} are subsampled according to the trick proposed in (Mikolov et al., 2013b). The optimal configurations of Skip-gram are: $k = 150$, $s = 5$, and $c = 10$.

Combining entity embeddings and word embeddings learnt by pTransE and Skip-gram respectively, "respectively" model can score all types of held-out triplets. For our jointly embedding model, we consider various alignment mechanisms and use equal numbers of threads for knowledge model and text model. The best configurations of "jointly" model are: $k = 150$, $s = 5$, $c = 10$, and $\alpha = 0.025$ which linearly decreases along with the 6 epochs of traversing text corpus.

Results. We first illustrate the comparison between TransE and pTransE over $e - e$ type triplets in Table 3. Observing the scores assigned to true triplets by TransE, we notice that triplets of popular relations generally have larger scores than those of rare relations. In contrast, pTransE, as a probabilistic model, assigns comparable scores to true triplets of both popular and rare relations. When we use a threshold to separate true triplets from false triplets of the same relation, there is no obvious difference between the two models. However, when all triplets are ranked together, assigning scores in a more uniform scale is definitely an advantage. Thus, the contradiction stems from the different training strategies of the two models and the consideration of relation-specific thresholds.

Classification accuracies over various types of held-out triplets are presented in Table 4. The "jointly" model outperforms the "respectively" model no matter which alignment mechanism(s) are used. Actually, for the "respectively" model, there is no interaction between entity embeddings and word embeddings during training and thus its predictions, over triplets that involve both entity and word at the same time, are not much better than random guessing. It is also a natural result that alignment by names is more effective than alignment by anchors. The number of anchors is much smaller than the number of overall chunks in our text corpus. In addition, the number of entities mentioned by anchors is very limited com-

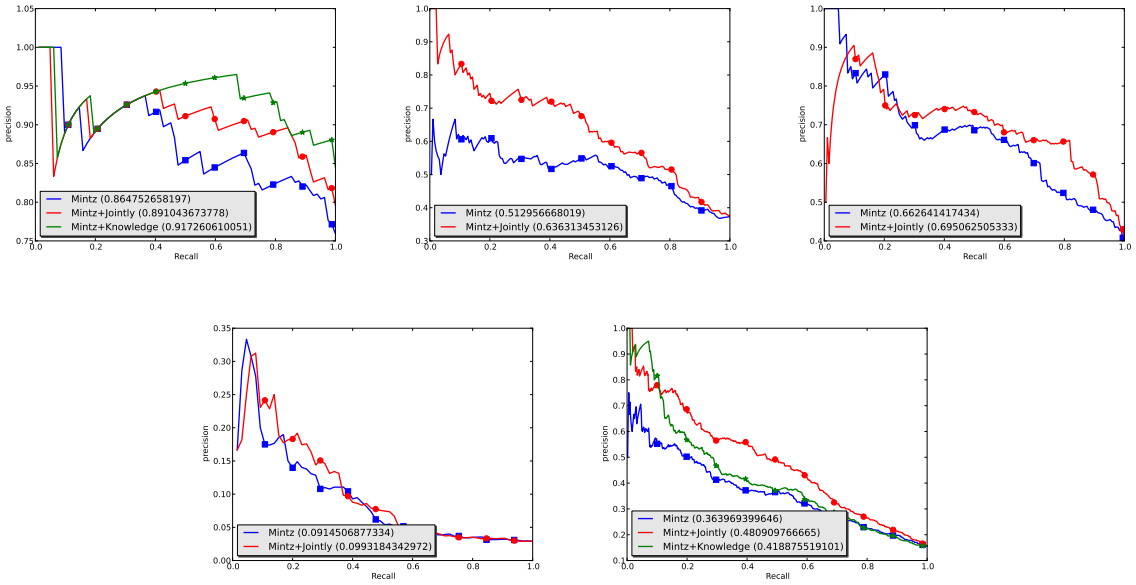

Figure 1: **Improving Relation Extraction**: PR curves of Mintz alone or combined with knowledge (pTransE) / jointly model over (a) $e-e$, (b) $w-e$, (c) $e-w$, (d) $w-w$, and (e) all triplets.

pared with $|\mathcal{E}|$. Thus, interactions brought in by anchors are not as significant as that of the name graph.

4.3 Improving Relation Extraction

It has been shown that embedding models are very complementary to extractors (Weston et al., 2013). However, some entities detected from text are out-of-kb entities. In such a case, triplets involving these entities cannot be handled by any existing knowledge embedding method, but our jointly embedding model can score them. As our model can cover more candidate triplets provided by extractors, it is expected to provide more significant improvements to extractors than any other embedding model. We confirm this point as follow.

Evaluation protocol. For relation extraction, we use a public dataset—NYT+FB (Riedel et al., 2010)[3], which distantly labels the NYT corpus by Freebase facts. We consider (Mintz et al., 2009) and Sm2r (Weston et al., 2013) as our extractors to provide candidate triplets as well as their plausibilities estimated according to text features.

For embedding, we first held out triplets from our training set that appear in the test set of NYT+FB. Then we train TransE, pTransE and the "jointly" model on the remaining training triplets as well as on our text corpus. Then we use these models to score each candidate triplet in the same way as the previous triplet classification experiment.

For combination, we first divide each candidate triplet into one of these categories: $e-e$, $e-w$, $w-e$, $w-w$, and "out-of-vocabulary". Because there is no embedding model that can score triplets involving out-of-vocabulary word/phrase, we just ignore these triplets.Please note that, for our jointly embedding model, there are no "out-of-vocabulary" triplets if we include the NYT corpus for training. We use the embedding models to score candidate triplets and combine the scores given by the embedding model with scores given by the extractors. For each type $e-e$, $e-w$, $w-e$, $w-w$ and their union (i.e. *all*), we rank the candidate triplets by their revisited scores and draw PR curve to observe which embedding method provides the most significant improvements to the extractors.

Implementation. For (Mintz et al., 2009), we use the implementation in (Surdeanu et al., 2012)[4]. We implement Sm2r by ourselves with the best hyperparameters introduced in (Weston et al., 2013). For TransE, pTransE, and the "jointly" model, we use the same implementations, scoring schemes, and optimal configurations as the triplet classification experiment.

To combine extractors with embedding mod-

[3] http://iesl.cs.umass.edu/riedel/ecml/

[4] http://nlp.stanford.edu/software/mimlre.shtml

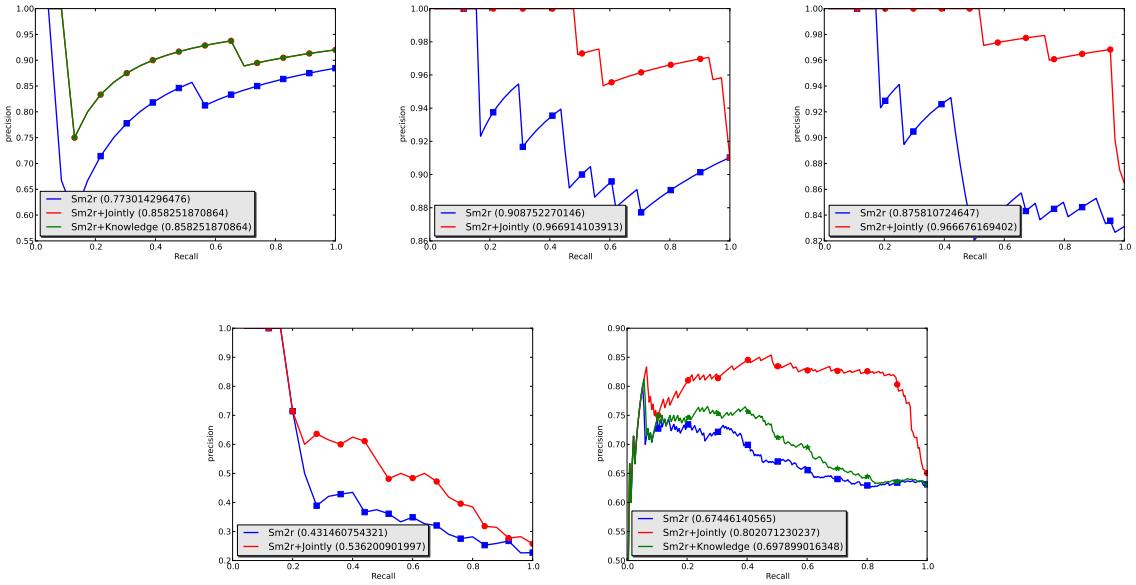

Figure 2: **Improving Relation Extraction**: PR curves of Sm2r alone or combined with knowledge (TransE) / jointly model over (a) $e - e$, (b) $w - e$, (c) $e - w$, (d) $w - w$, and (e) all triplets.

els, we consider two schemes. Since Mintz scores candidate triplets in a probabilistic manner, we linearly combine its scores with the scores given by pTransE or the "jointly" model: $\beta \Pr_{\text{Mintz}} + (1-\beta) \Pr_{\text{pTransE/Jointly}}$ where β is enumerated from 0 to 1 with 0.025 as a search step. On the other hand, neither Sm2r nor TransE is a probabilistic model. Thus, we combine Sm2r with TransE or the "jointly" model according to the scheme proposed in (Weston et al., 2013) where for each candidate (h, r, t), if $\sum_{r' \neq r} \delta(\text{Score}(h, r, t) < \text{Score}(h, r', t))$ is less than τ, we increase $\text{Score}_{Sm2r}(h, r, t)$ by p. We search for the best β, τ, and p on another dataset—Wikipedia corpus distantly labeled by Freebase.

Result. We present the PR curves in Fig. (1, 2). Over candidate triplets provided by either Mintz or Sm2r, the "jointly" model is consistently comparable with the "knowledge" model (TransE/pTransE) over $e - e$ triplets while it outperforms the "knowledge" model by a considerable margin over triplets of other types. These results confirm the advantage of jointly embedding and are actually straightforward results of our triplet classification experiment because the only difference is that the triplets here are provided by the extractor.

Table 6: **Phrases Analogical Reasoning Task**.

Method	Accuracy (%)	Hits@10 (%)
Skip-gram	18.0	56.1
Jointly (anchor)	**27.6**	**65.0**
Jointly (name)	11.3	40.6
Jointly (anchor+name)	18.3	54.0

Table 7: **Constructed Analogical Reasoning Task**.

Method	Accuracy (%)	Hits@10 (%)
Skip-gram	10.5	14.1
Jointly (anchor)	10.5	14.3
Jointly (name)	11.5	16.2
Jointly (anchor+name)	**11.6**	**16.5**

4.4 Analogical Reasoning Task

We compare our method with Skip-gram on this task to observe and study the influences of both knowledge embedding and alignment mechanisms on the quality of word embeddings.

Evaluation protocol. We use the same public datasets as in (Mikolov et al., 2013b): 19,544 word analogies[5]; 3,218 phrase analogies[6]. We also construct analogies from our held-out triplets (see Table 2) by first concatenating two entity pairs of the same relation to form an analogy and

[5] code.google.com/p/word2vec/source/browse/trunk/questions-words.txt

[6] code.google.com/p/word2vec/source/browse/trunk/questions-phrases.txt

Table 5: **Words Analogical Reasoning Task**.

Method	Accuracy (%)			Hits@10 (%)		
	Semantic	Syntactic	Total	Semantic	Syntactic	Total
Skip-gram	71.4	69.0	70.0	90.4	89.3	89.8
Jointly (anchor)	75.3	68.3	**71.2**	91.5	88.9	**89.9**
Jointly (name)	54.5	54.2	59.0	75.8	86.5	82.1
Jointly (anchor+name)	56.5	65.7	61.9	78.1	87.6	83.6

then replacing the entities by corresponding entity names, e.g., "(Obama, Honolulu, David Beckham, London)" where the relation is "Born-in".

Following (Mikolov et al., 2013b), we only consider analogies that consist of the top-K most frequent words/phrases in the vocabulary. For each analogy denoted by (h_1, t_1, h_2, t_2), we enumerate all the top-K most frequent words/phrases w except for h_1, t_1, h_2, and calculate the distance (Cosine/Euclidean according to specific model) between $\mathbf{h}_2 + (\mathbf{t}_1 - \mathbf{h}_1)$ and $\mathbf{w}$. Ordering all these words/phrases by their distances in ascending order, we obtain the rank of the correct answer t_2. Finally, we report *Hits@10* (i.e., the proportion of correct answers whose ranks are not larger than 10) and *accuracy* (i.e., Hits@1). For word analogies and constructed analogies, we set $K = 200,000$; while for phrase analogies, we set $K = 1,000,000$ to recall sufficient analogies.

Implementation. For Skip-gram and the "Jointly" (anchor/name/anchor+name) model, we use the same implementations and optimal configurations as the triplet classification experiment.

Results. Jointly embedding using Wikipedia anchors for alignment consistently outperforms Skip-gram (Table 5, 6, 7) showing that the influence of knowledge embedding, injected into word embedding through Wikipedia anchors, is beneficial. The vector of an ambiguous word is often a mixture of its several meanings but, in a specific context, the word is disambiguated and refers to a specific meaning. Using global word embedding to predict words within a specific context may pollute the embeddings of surrounding words. Alignment by anchors enables entity embeddings to alleviate the propagation of ambiguities and thus improves the quality of word embeddings.

Using entity names for alignment hurts the performance of analogies of words and phrases (Table 5, 6). The main reason is that these analogies are popular facts frequently mentioned in text while a name graph forces word embeddings to satisfy both popular and rare facts. Another reason stems from the versatility of mentioning an entity. Consider "(Japan, yen, Europe, euro)" for example. Knowledge embedding is supposed to give significant help to completing this analogy as "/location/country/currency"$\in \mathcal{R}$. However, the entity of Japanese currency is named "Japanese yen" rather than "yen" and thus the explicit translation learnt from knowledge embedding is not directly imposed on the word embedding of "yen". In contrast, using entity names for alignment improves the performances on constructed analogies (Table 7). Since there is a relation $r \in \mathcal{R}$ for each constructed analogy $(w_{h_1}, w_{t_1}, w_{h_2}, w_{t_2})$, although neither (w_{h_1}, r, w_{t_1}) nor (w_{h_2}, r, w_{t_2}) is present in the name graph, other facts involving these words act on the vectors of these words, in the same manner of traditional knowledge embedding.

Overall, any high-quality entity linking system can be used to further improve the performance.

5 Conclusions

In this paper, we introduced a novel method of jointly embedding knowledge graphs and a text corpus so that entities and words/phrases are represented in the same vector space. In such a way, our method can perform prediction on any candidate facts between entities/words/phrases, going beyond previous knowledge embedding methods, which can only predict facts whose entities exist in knowledge graph. Extensive, large-scale experiments show that the proposed method is very effective at reasoning new facts. In addition, we also provides insights into word embedding, especially on the capability of analogical reasoning. In this aspect, we empirically observed some hints that jointly embedding also helps word embedding.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

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, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 301–306.

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.

Kai-Wei Chang, Wen-tau Yih, and Christopher Meek. 2013. Multi-relational latent semantic analysis. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1602–1612, Seattle, Washington, USA, October. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Miao Fan, Deli Zhao, Qiang Zhou, Zhiyuan Liu, Thomas Fang Zheng, and Edward Y. Chang. 2014. Distant supervision for relation extraction with matrix completion. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 839–849, Baltimore, Maryland, June. Association for Computational Linguistics.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

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.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465. 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.

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 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1366–1371, Seattle, Washington, USA, October. Association for Computational Linguistics.

Xingxing Zhang, Jianwen Zhang, Junyu Zeng, Jun Yan, Zheng Chen, and Zhifang Sui. 2013. Towards accurate distant supervision for relational facts extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 810–815, Sofia, Bulgaria, August. Association for Computational Linguistics.