Neural Knowledge Acquisition from Joint Representation of Knowledge Graph and Text

Abstract

Joint learning methods in Natural Language Processing (NLP) have usually focused on some similar tasks so that complementary linguistic features of morphology, syntax and semantics from different models would benefit each other. In this paper, we propose a novel joint learning framework for both knowledge graph completion (KGC) and relation extraction (RE). By learning representations of KGs and text within a unified semantic space, structural KG features and rich flexible textual features enhance models together. The joint mechanism enables us to take both KGs and plain text into consideration at the same time and perform KGC and RE more accurately. In experiments, we evaluate our joint learning model on three classical tasks including relation extraction, entity prediction and relation prediction. The experiment results show that our joint model can significantly and consistently improve the performances on the three tasks as compared with other baselines without joint learning. The source code of our joint model would be released in the public then.

1 Introduction

People construct various large-scale knowledge graphs (KGs) to organize structural knowledge about the world, such as WordNet (Miller, 1995), YAGO (Suchanek et al., 2007), DBPedia (Auer et al., 2007), Freebase (Bollacker et al., 2008), Knowledge Vault (Dong et al., 2014) and Wikidata(Vrandečić and Krötzsch, 2014). A typical knowledge graph is a multiple-relational directed graph with nodes corresponding to entities, and edges corresponding to relations between these entities. The facts in KGs are usually recorded as a set of relational triples (h, r, t) with h, t indicating *head*, *tail* entities and r indicating the relation between h and t, e.g., (*Mark Twain*, `PlaceOfBirth`, *Florida*). Owing to rich and structural inner information, KGs are playing an important role in numerous applications especially in question answering and web searching.

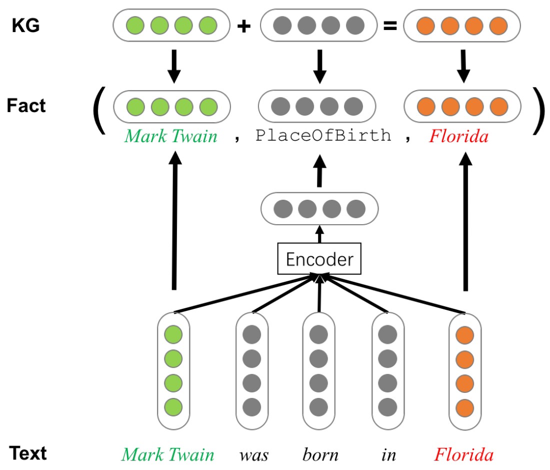

Figure 1: The framework of joint representation learning of the knowledge graph and text.

However, typical large-scale KGs are usually far from complete. The task of knowledge graph completion (KGC) aims to enrich KGs with novel facts. Based on the network structure of KGs, many graph-based methods have been proposed to mine novel facts between entities (Lao et al., 2011; Lao and Cohen, 2010). Many efforts are also devoted to extract relational facts from plain text (Surdeanu et al., 2012; Riedel et al., 2013; Min et al., 2013; Zeng et al., 2014, 2015; dos Santos et al., 2015; Lin et al., 2016) for the task of relation extraction (RE). Though the two tasks are similar, the past approaches cannot jointly take both KGs and plain text into consideration.

In recent years, neural-based knowledge representation has been proposed to encode both entities and relations into a low-dimensional space, which are capable to find novel facts (Bordes et al., 2013; Wang et al., 2014b; Lin et al., 2015; Ji et al., 2015; He et al., 2015; Xiao et al., 2015; Ji et al., 2016). More importantly, neural models enable us to conduct joint representation learning of KGs and text within a unified semantic space, and perform KGC and RE more accurately.

Some pioneering works have been done. For example, (Wang et al., 2014a) performs joint learning simply considering alignment between words and entities, and (Toutanova et al., 2015) extracts textual relations from plain text using dependency parsing to enhance relation embeddings. These works either consider only partial information in plain text (only entity mentions in (Wang et al., 2014a; Xie et al., 2016; Wu et al., 2016; Zeng et al., 2016) and only textual relations in (Toutanova et al., 2015)), or rely on complicated linguistic analysis (dependency parsing in (Toutanova et al., 2015)) which may bring inevitable parsing errors.

To address these issues, we propose a novel framework for joint representation learning. As shown in Figure 1, the framework is expected to take full advantages of both KGs and text via complicated alignments with respect to words, entities and relations. Moreover, our method applies deep neural networks with a knowledge-attention mechanism instead of native linguistic analysis to encode the semantics of sentences, which is especially capable of modeling large-scale and noisy Web text.

We conduct experiments on a real-world dataset whose KG extracted from Freebase and text derived from the New York Times corpus. We evaluate our method on the tasks of KGC (entity prediction and relation prediction) and RE (relation extraction from text). Experiment results demonstrate that, our method can effectively perform joint representation learning and obtain more informative knowledge and text representation, which significantly outperforms other baseline methods.

2 Related Work

The work in this paper relates to representation learning of KGs, words, textual relations and neural networks with attention. Related works are reviewed as follows.

Representation Learning of KGs. A variety of approaches have been proposed to encode both entities and relations into a continuous low-dimensional space. Inspired by (Mikolov et al., 2013b), TransE (Bordes et al., 2013) regards the relation r in each fact (h, r, t) as a translation from h to t within the low-dimensional space, i.e., $\mathbf{h}+\mathbf{r}=\mathbf{t}$, where $\mathbf{h}$ and $\mathbf{t}$ are entity embeddings and $\mathbf{r}$ is relation embedding. Despite of its simplicity, TransE achieves the state-of-the-art performance of representation learning for KGs, especially for those large-scale and sparse KGs. Hence, we simply incorporate TransE in our method to handle representation learning for KGs.

Note that, our method is also flexible to incorporate extension models of TransE, such as TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015) and TranSparse (Ji et al., 2016). which is not the focus of this paper and will be left as our future work.

Representation Learning of Words. Given a text corpus, we can learn word representations without supervision. The learning objective is defined as the likelihood of predicting its context words of each word or vice versa (Mikolov et al., 2013b). Continuous Bag-of-Words (CBOW) (Mikolov et al., 2013a) and Skip-Gram (Mikolov et al., 2013c) are state-of-the-art methods for word representation learning. The learned word embeddings can capture both syntactic and semantic features of words derived from plain text corpus. As reported in many previous works, deep neural network will benefit significantly if being initialized with pre-trained word embeddings (Erhan et al., 2010). In this work, we apply Skip-Gram for word representation learning, which serves as initialization for joint representation learning of text and KGs.

Representation Learning of Textual Relations. Many works aim to extract relational facts from large-scale text corpus (Mintz et al., 2009; Riedel et al., 2010). This indicates textual relations between entities are contained in plain text. In recent years, deep neural models such as convolutional neural networks (CNN) have been proposed to encode semantics of sentences to identify relations between entities (Zeng et al., 2014; dos Santos et al., 2015; Zeng et al., 2015; Lin et al., 2016). As compared to conventional models, neural models are capable to accurately capture textual relations between entities from text sequences without explicitly linguistic analysis, and further encode into continuous vector space. Hence, in this work we apply CNN to embed textual relations and conduct joint learning of text and KGs with respect to relations.

Many neural models such as recurrent neural networks (RNN) (Zhang and Wang, 2015) and long-short term memory networks (LSTM) (Xu et al., 2015; Miwa and Bansal, 2016) have also been explored for RE. These models can also be applied to perform representation learning for textual relations. However, shortcomings of RNN in long-term dependencies and time efficiency make us prefer CNN for representation of textural relations.

Neural Networks with Attention. (Bahdanau et al., 2014) first proposed the attention mechanism in neural networks for machine translation. The attention mechanism is implemented to select the most relevant and important hidden features encoded from the original sentence to better decode the target sentence. Due to the ability to catch key points in text, many attention models has been proposed for several NLP tasks, such as entity type classification (Shimaoka et al., 2016), machine reading (Hermann et al., 2015; Dhingra et al., 2016; Sordoni et al., 2016). (Yang et al., 2016) proposed a hierarchical attention network over sentences for document classification, and further more, (Lin et al., 2016) use the sentence selective attention model for RE. In this paper, we propose a multilevel-attention mechanism for both words and sentences under knowledge KG guidance.

3 The Framework

In this section we introduce the framework of joint representation learning, starting by notations and definitions.

3.1 Notations and Definitions

We denote a knowledge graph as $G = \{E, R, T\}$, where E indicates a set of entities, R indicates a set of relation types, and T indicates a set of fact triples. Each triple $(h, r, t) \in T$ indicates there is a relation $r \in R$ between $h \in E$ and $t \in E$.

We denote a text corpus as D and its vocabulary as V, containing all words, phrases and entity mentions. In the corpus D, each sentence is denoted as a word sequence $s = \{x_1, \ldots, x_n\}, x_i \in V$, and n is the sentence length.

For entities, relations and words, we use the bold face to indicate their corresponding low-dimensional vectors. For example, the embeddings of $h, t \in E$, $r \in R$ and $x \in V$ are $\mathbf{h}, \mathbf{t}, \mathbf{r}, \mathbf{x} \in \mathbb{R}^{k_w}$ respectively, where k_w is the embedding dimension.

3.2 Joint Learning Method

As mentioned in Section 2, representation learning methods have been proposed for knowledge graphs and text corpora respectively. In this work, we propose a joint learning framework for both the KG and text.

In this framework, we aim to jointly learn representations of entities, relations and words. With denoting all these representations as model parameters $\theta = \{\theta_E, \theta_R, \theta_V\}$, the framework aims to find optimized parameters

$$\hat{\theta} = \arg\max_{\theta} P(G, D|\theta), \tag{1}$$

where $\theta_E, \theta_R, \theta_V$ are parameters for entities, relations and words respectively. $P(G, D|\theta)$ is the conditional probability defined over the knowledge graph G and the text corpus D given the parameters θ. The conditional probability can be further decomposed as:

$$P(G, D|\theta) = P(G|\theta_E, \theta_R)P(D|\theta_V) \tag{2}$$

$$= \prod_{(h,r,t)\in G} P((h, r, t)|\theta_E, \theta_R) \prod_{s\in D} P((s, r_s)|\theta_V),$$

where $P((h, r, t)|\theta_E, \theta_R)$ denotes the conditional probability of relational triples (h, r, t) in the knowledge graph G and $P((s, r_s)|\theta_V)$ denotes the conditional probability of sentences and their corresponding textual relations (s, r_s) in the text corpus D.

$P(G|\theta_E, \theta_R)$ is responsible to learn representations of both entities and relations from the knowledge graph G. This part will be introduced in detail in Section 3.3.

$P(D|\theta_V)$ is responsible to learn representations of sentence words as well as textual relations from the text corpus D. It is straightforward to learn word representations from text as discussed in

Section 2. Since entities and relations are not explicitly shown in text, we have to identify entities and relations in text to support information exchange between entities and words, relations and textual relations. The process is realized by entity-text alignment and relation-text alignment.

Entity-Text Alignment. Many entities are mentioned in text. Due to the complex polysemy of entity mentions (e.g., an entity name Washington in a sentence could be indicating either a person or a location), it is non-trivial to build entity-text alignment. The alignment can be built via entity linking techniques or anchor text information. In this paper, we simply use the anchor text annotated in articles to build the alignment between entities in E and entity mentions in V. We will share the aligned entity representations in θ_E to corresponding entity mentions in θ_V.

Relation-Text Alignment. As mentioned in Section 2, textual relations can be extracted from text. Hence, relation representation can also be learned from plain text. Inspired by the idea of distant supervision, for a relation $r \in R$, we collect all entity pairs $Pair_r = \{(h,t)\}$ connected by r in the KG. Afterwards, for each entity pair in $Pair_r$, we extract all sentences that contain the both entities from D, and regard them as the positive instances of the relation r. We can further apply deep neural networks to encode the semantic of these sentences into the corresponding relation representation. The process will be introduced in detail in Section 3.4.

In summary, the framework enables joint representation learning of both entities and relations by taking full advantages of both KG and text. The learned representations are expected to be more informative and robust, which will be verified in experiments.

3.3 Representation Learning of KGs

We aim to embed entities and relations to capture the correlations between them. When learning from relational triples, we usually optimize the conditional probability $P(h|(r,t),\theta_E,\theta_R)$, $P(t|(h,r),\theta_E,\theta_R)$, and $P(r|(h,t),\theta_E,\theta_R)$ instead of $P((h,r,t)|\theta_E,\theta_R)$

For each entity pair (h,t) in a KG G, we define their latent relation embedding $\mathbf{r}_{ht}$ as a translation from $\mathbf{h}$ to $\mathbf{t}$, which can be formalized as:

$$\mathbf{r}_{ht} = \mathbf{t} - \mathbf{h}. \quad (3)$$

Meanwhile, each triple $(h,r,t) \in T$ has an explicit relation r between h and t. Hence, we can define the scoring function for each triple as follows:

$$f_r(h,t) = b1 - \|\mathbf{r}_{ht} - \mathbf{r}\|_2 = b1 - \|(\mathbf{t} - \mathbf{h}) - \mathbf{r}\|_2. \quad (4)$$

where b1 is a bias constant. This indicates that, for each triple (h,r,t) in T, we expect $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$.

Based on the above scoring function, the conditional probability can be formalized over all triples in T as follows:

$$P(h|(r,t),\theta_E,\theta_R) = \frac{exp(f_r(h,t))}{\sum_{h' \in E} exp(f_r(h',t))} \quad (5)$$

We also define $P(t|(h,r),\theta_E,\theta_R)$ and $P(r|(h,t),\theta_E,\theta_R)$ in the same way by choosing corresponding normalization terms respectively. In fact, this representation objective is consistent with TransE (Bordes et al., 2013). Other knowledge representation (KR) models evolved from TransE can also be easily adopted in this framework by changing the scoring function, such as TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015) and TranSparse (Ji et al., 2016).

3.4 Representation Learning of Textual Relations

Given a sentence containing two entities, the words in the sentence usually expose implicit features of the textual relation between the two entities. As shown in (Zeng et al., 2014; Toutanova et al., 2015; Lin et al., 2016), the textual relations can be learned with deep neural networks and encoded in the low-dimensional semantic space.

We follow (Zeng et al., 2014; Toutanova et al., 2015; Lin et al., 2016) and apply convolutional neural networks (CNN) to model textual relations from text.

3.4.1 Overall Architecture

Figure 2 depicts the overall architecture of CNN for modeling textual relations. For a sentence s containing (h,t) with a relation r_s, the architecture takes word embeddings $\mathbf{s} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of the sentence s as input, and after passing through two layers within CNN, outputs the embedding of the textual relation $\mathbf{r}_s$. Our method will further

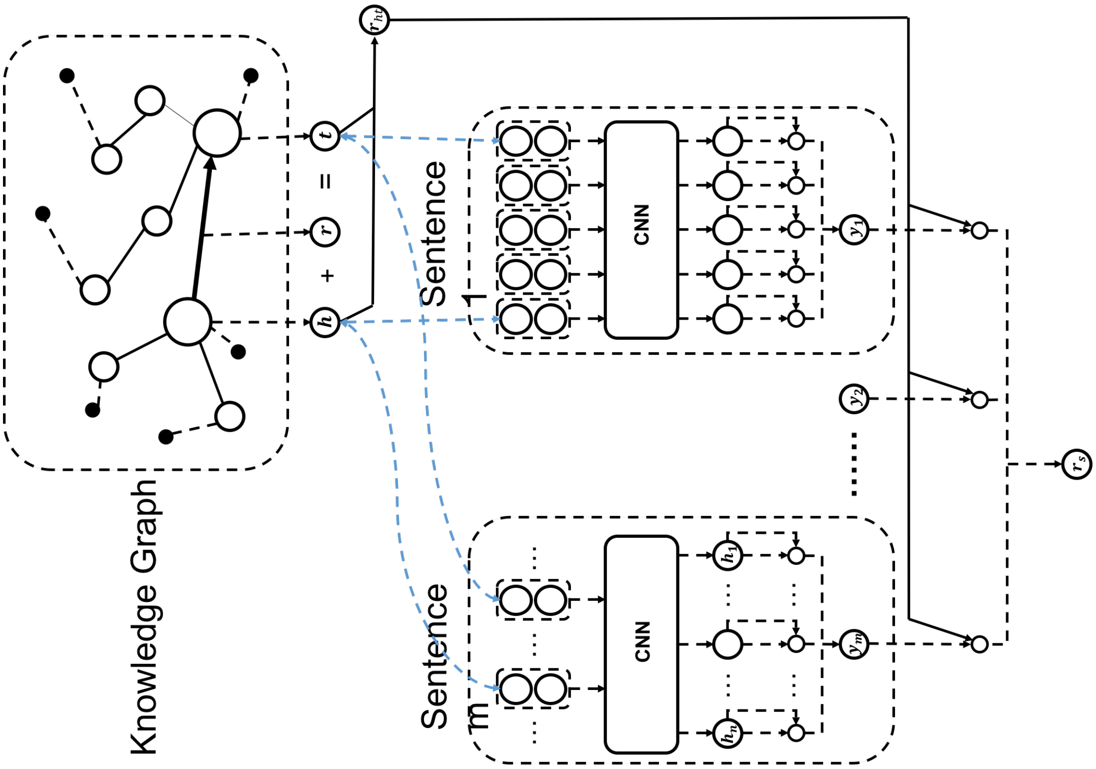

Figure 2: Convolutional neural networks for representation learning of textual relations.

learn to get the scoring function through a softmax layer as follows,

$$f(s) = \mathbf{M}\mathbf{r}_s + \mathbf{d}, \qquad (6)$$

where $\mathbf{d} \in \mathbb{R}^{\|R\|}$ is a bias vector and $\mathbf{M}$ is the representation matrix to calculate the relation scorings. Finally we define the conditional probability $P((s, r_s)|\theta_V)$

$$P((s, r_s)|\theta_V) = \frac{exp(f_{r_s}(s))}{\sum_{r \in R} exp(f_r(s))} \qquad (7)$$

In this paper, our CNN contains an input layer, a convolution layer and a multilevel-attention mechanism. All of these are introduced in detail as follows.

3.4.2 Input Layer

Given a sentence s made up of n words $s = \{x_1, \ldots, x_n\}$, the input layer transforms the words of s into corresponding word embeddings $\mathbf{s} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. For a word x_i in the given sentence, its input embedding $\mathbf{x}_i$ is composed of two real-valued vectors: its textual word embedding $\mathbf{w}_i$ and its position embedding $\mathbf{p}_i$.

Textual word embeddings encode the semantics of the corresponding words, which are usually pre-trained from plain text via word representation learning, as introduced in Section 3.5.

Word position embeddings (WPE) is originally proposed in (Zeng et al., 2014). WPE is a position feature indicating the relative distances of the given word to the marked entities in the sentence. As shown in Figure 2, the relative distances of the word *born* to the entities *Mark Twain* and *Florida* are -2 and $+2$ respectively. We map each distance to a vector of dimension k_p in the continuous latent space. Given the word x_i in the sentence s, the word position embedding is $\mathbf{p}_i = [\mathbf{p}_i^h, \mathbf{p}_i^t]$, where $\mathbf{p}_i^h$ and $\mathbf{p}_i^t$ are vectors of distances to the head entity and tail entity respectively.

We simply concatenate textual word embeddings and word position embeddings to build the input for CNN:

$$\mathbf{s} = \{[\mathbf{w}_1; \mathbf{p}_1], \ldots, [\mathbf{w}_n; \mathbf{p}_n]\}. \qquad (8)$$

3.4.3 Convolution Layer

By taking $\mathbf{s}$ as the input, the convolution layer will output $\mathbf{y}$. The generation process is formalized as follows.

We slide a window of size m over the input word sequence. For each move, we can get an embedding $\mathbf{x}'_i$ as:

$$\mathbf{x}'_i = \left[\mathbf{x}_{i-\frac{m-1}{2}}; \ldots; \mathbf{x}_i; \ldots; \mathbf{x}_{i+\frac{m-1}{2}}\right], \quad (9)$$

which is obtained by concatenating m vectors in s with $\mathbf{x}_i$ as center. For instance in Figure 2, a window slides through the input vectors $\mathbf{s}$ and concatenates every three word embeddings. Afterwards, we transform $\mathbf{x}'_i$ into the hidden layer vector $\mathbf{h}_i$

$$\mathbf{h}_i = \tanh(\mathbf{W}\mathbf{x}'_i + \mathbf{b}), \quad (10)$$

where $\mathbf{W} \in \mathbb{R}^{k_c \times mk_w}$ is the convolution kernel, $\mathbf{b} \in \mathbb{R}^{k_c}$ is a bias vector, k_c is the dimension of hidden layer vectors $\mathbf{h}_i$, k_w is the dimension of input vectors $\mathbf{x}_i$, and m is the window size.

3.4.4 Attention Structure

In view of some unique features over words and sentences, we add the multilevel-attention to our neural network model, which can recognize informative parts and make the model more robust. Because of this, we implement attention over both word and sentence level to extract more features from text in our work.

Word Attention. It is obvious that not all words contribute equally to the representation of the textual relations. thus, we use the word level attention mechanism to highlight such words that are important to the textual relations. After the convolution layer operation, we have some hidden layer vector $\mathbf{h}_1, \ldots, \mathbf{h}_n$. With the corresponding textual relation, we can get the weight for each hidden layer vector with a two-layer feed forward neural networks whose weight matrices are $\mathbf{W}_w \in \mathbb{R}^{k_a k_c}$ and $\mathbf{A}_w \in \mathbb{R}^{k_a}$, k_a is the dimension of word attention layer vectors. Specifically,

$$\mathbf{e}_j = tanh(\mathbf{W}_w \mathbf{h}_j + \mathbf{b}_w) \quad (11)$$

$$a_j = \frac{exp(\mathbf{A}_w \cdot \mathbf{e}_j)}{\sum_{k=1}^{n} exp(\mathbf{A}_w \cdot \mathbf{e}_k)} \quad (12)$$

$$\mathbf{y} = \sum_{j=1}^{n} a_j \mathbf{h}_j \quad (13)$$

where the normalized attention value a_j is the weight for jth hidden layer vector $\mathbf{h}_j$. We take a weighted sum of the hidden layer vectors for the single sentence textual relation representation by the word attention.

Sentence Knowledge Attention. Above the word attention layer, we add second-step attention over the sentence level. For some $(h, r, t) \in T$, there may be several sentences containing (h, t) with a relation r. These sentences' single textual relation representation are $\mathbf{y}_1, \ldots, \mathbf{y}_m$, where m is the number of sentences. We argue that some sentences contribute more to the finial textual relation representation. To highlight these sentences, we use latent relation embedding $\mathbf{r}_{ht} \in \mathbb{R}^{k_w}$ as sentence knowledge attention and $\mathbf{W}_s \in \mathbb{R}^{k_w k_c}$ as weight matrices:

$$\mathbf{e}_j = tanh(\mathbf{W}_s \mathbf{y}_j + \mathbf{b}_s) \quad (14)$$

$$a_j = \frac{exp(\mathbf{r}_{ht} \cdot \mathbf{e}_j)}{\sum_{k=1}^{m} exp(\mathbf{r}_{ht} \cdot \mathbf{e}_k)} \quad (15)$$

$$\mathbf{r_s} = \sum_{j=1}^{m} a_j \mathbf{y}_j \quad (16)$$

where the normalized attention value a_j is the weight for jth sentence vector $\mathbf{y}_j$. We take a weighted sum of sentence vectors for the finial global textual relation representation $\mathbf{r_s}$. After we get $\mathbf{r}$, we can use the textual relation embedding for the scoring function Eq. (6).

3.5 Initialization and Implementation Details

Here we introduce the learning and optimization details for our joint model. We define the optimization function as the log-likelihood of the objective function in Eq. 2,

$$\begin{aligned} \mathcal{L}_\theta(G, D) &= \log(P(G, D|\theta)) + \lambda\|\theta\|_2 \quad (17) \\ &= \log(P(G|\theta_E, \theta_R) + \log(P(D|\theta_V)) + \lambda\|\theta\|_2 \end{aligned}$$

where λ is a harmonic factor, and $\|\theta\|_2$ is the regularizer defined as L_2 distance.

There are a large number of parameters to be optimized for joint learning. It is thus crucial to initialize these parameters appropriately. For those aligned entities and words, we initialize their embeddings via word representation learning. We follow (Mikolov et al., 2013c) and use Skip-Gram to learn word representations from the given text corpus. For relations and other entities, we initialize their embeddings randomly.

Both the knowledge model and textual relation model CNN are optimized simultaneously using stochastic gradient descent (SGD). The parameters of all models are trained using a batch training algorithm. Note that, the gradients of CNN parameters will be back-propagated to the input word embeddings so that the embeddings of both entities and words can also be learned from plain text via CNN.

4 Experiments

For knowledge graph completion (KGC) we conduct experiments on link prediction containing entity prediction and relation prediction. For relation extraction (RE) we conduct experiments on textual relations extraction from sentences. We evaluate the performance of our joint model with various single baselines.

4.1 Experiment Settings

4.1.1 Datasets

Knowledge Graph. We select Freebase (Bollacker et al., 2008) as the knowledge graph for joint learning. Freebase is a widely-used large-scale world knowledge graph. In this paper, we adopt datasets extracted from Freebase, FB15K and FB40K in our experiments. FB15K has been used as the benchmark for link prediction (Bordes et al., 2013; Wang et al., 2014b; Lin et al., 2015; Ji et al., 2015; He et al., 2015; Xiao et al., 2015; Ji et al., 2016). FB40K has been used as the benchmark for relation extraction (Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012; Zeng et al., 2014, 2015; Lin et al., 2016). FB15K contains 14,951 entities, 1,345 relations and 592,213 facts. FB40K contains 69,512 entities, 1,324 relations and 335,350 facts after expanding.

Text Corpus. We select sentences from the New York Times articles to align with KGs for joint learning. To ensure alignment accuracy, we only consider those sentences with anchor text linking to the entities in KGs. We extract $194,385$ sentences containing both head and tail entities in FB15K triples, and annotate with the corresponding relations in triples. The sentences are labeled with $47,103$ FB15K triples, including 699 relations and 6053 entities. We name the corpus as NYT-FB15K. The sentences for FB40K come from the dataset[1] which is developed by (Riedel et al., 2010), containing $570,088$ sentences, $63,696$ entities, 56 relations and $293,175$ facts. We name the corpus as NYT-FB40K.

Table 1: Parameter settings

Harmonic Factor λ	0.001
Knowledge Learning Rate α_k	0.001
Text Learning Rate α_t	0.01
Hidden Layer Dimension d_c	230
Attention Layer Dimension d_a	230
Word/Entity/Relation Dimension d_w	50
Position Dimension d_p	5
Window Size m	3
Dropout Probability p	0.5

4.1.2 Evaluation Tasks

In experiments we evaluate the joint learning model and baselines with three parts:

(1) **Entity Prediction.** The task aims at predicting missing entities in a triple according to the embeddings of another entity and relation.

(2) **Relation Prediction.** The task aims at predicting missing relations in a triple according to the embeddings of head and tail entities.

(3) **Relation Extraction.** We are also interested in extracting relational facts between novel entities not included in KGs. Hence, we conduct relation extraction from sentences in text.

4.1.3 Parameter Settings

In our joint model, we select the learning rate α_k on the knowledge side among $\{0.1, 0.01, 0.001\}$, and learning rate α_t on the text side among $\{0.1, 0.01, 0.001\}$. The sliding window size $\mathbf{m}$ is among $\{3, 5, 7\}$ and embedding dimension $\mathbf{k_w}$ is among $\{50, 100, 150\}$. For other parameters, since they have little effect on the results, we follow the settings used in (Zeng et al., 2014; Lin et al., 2016) so that we can compare joint learning results and single learning results. In Table 1 we show all parameters used in the experiments.

4.2 Results of Relation Extraction

The task aims to extract relational facts from plain text. Most models (Mintz et al., 2009; Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012; Zeng et al., 2014, 2015; Lin et al., 2016) take knowledge graphs as distant supervision to automatically annotate sentences in text corpora as training instances, and then extract textual features to build relation classifiers. Since there is much noise in plain text and distant supervision, it makes the task not easy. With this task, we want to investigate the effectiveness of our joint model for learning CNN models.

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

We follow (Weston et al., 2013) to conduct evaluation. The evaluation construct candidate triples combined by entity pairs in testing set and various relations, ask systems to rank these triples according to the corresponding sentences of entity pairs, and by regarding the triples in knowledge graphs as correct and others as incorrect, evaluate systems with precision-recall curves.

The evaluation results on NYT-FB40K test set are shown in Figure 3, where "Joint" indicates the CNN model learned jointly in our model, and "CNN+ONE" indicates the conventional CNN model learned individually from plain text without attention. "CNN+ATT" indicates the conventional CNN model with sentence attention proposed by (Lin et al., 2016), which is the state-of-the-art method for RE.

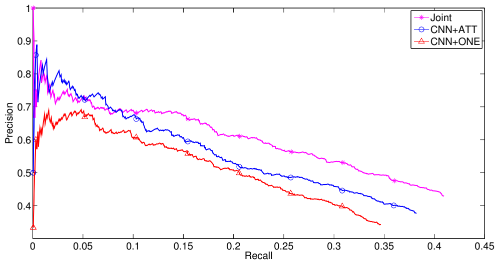

Figure 3: Aggregate precision/recall curves of joint learning CNN and single learning CNN.

We also compare our joint model with feature-based approaches, such as Mintz (Mintz et al., 2009), MultiR (Hoffmann et al., 2011) and MIML (Surdeanu et al., 2012). Mintz is a traditional distant supervised model, MultiR is a probabilistic, graphical model of multi-instance learning and MIML is a jointly model for both multiple instances and multiple relations. The results are shown in Figure 4

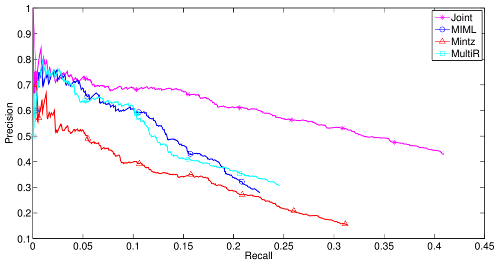

Figure 4: Performance comparison of joint model and traditional methods.

From the results we observe that:

(1) The joint model get much higher prediction accuracies on the whole than CNN trained individually. It demonstrates that the joint model successfully makes use of KGs to train CNN for relation extraction. Compared to single training CNN with attention mechanism, the joint model has 5% to 10% improvements. These models for sentence encoding are similar, the difference of these methods are the sentence attention and the joint process. Hence, it shows that utilizing joint learning along with knowledge attention make the single model just with the sentence attention more robust. When compared to single training CNN without attention mechanism, the joint model has more amazing 10% to 20% improvements.

(2) Although CNN with sentence attention can achieve better performance than the joint model when the recall is small, the Joint model still significantly outperforms CNN over all the range. Features come from KGs are efficient but smooth. When knowledge features are injected into CNN via joint learning, the overall effect becomes much better along with little discrimination loss.

(3) Compared with feature-based approaches, the joint model significantly outperforms all feature-based methods over the entire range of recall. The performance of feature-based method drop much more faster, however, the joint model has a reasonable precision even when the recall reaches 0.4. It demonstrates that the human-designed features are limiting, though these features are structured. The knowledge features are also structured and learned by the model automatically without supervision. Hence, proposing better models to mining features is more important than designing new features for RE.

4.3 Results of Link Prediction

4.3.1 Entity Prediction

Entity prediction has also been used for evaluation in (Bordes et al., 2013; Wang et al., 2014b; Lin et al., 2015; Ji et al., 2015; He et al., 2015; Xiao et al., 2015; Ji et al., 2016). More specifically, we need to predict the tail entity when given a triple $(h, r, ?)$ or predict the head entity when given a triple $(?, r, t)$. In this task, for each missing entity, the system is asked to rank all candidate entities from the knowledge graph instead of only giving one best result. For each test triple (h, r, t), we replace head and tail entities with all entities in FB15K ranked in descending order of similarity

Metric	Predicting Head				Predicting Tail				Overall	
	1-to-1	1-to-N	N-to-1	N-to-N	1-to-1	1-to-N	N-to-1	N-to-N	Triple Avg.	Relation Avg.
SE	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3	39.8	-
SME	35.1	69.6	19.9	40.3	32.7	14.9	76.0	43.3	41.3	-
TransE	43.7	65.7	18.2	47.2	43.7	19.7	66.7	50.0	47.1	-
TransH	66.8	87.6	30.2	64.5	65.5	39.8	83.3	67.2	64.4	-
TransR	78.8	89.2	38.1	66.9	79.2	38.4	**90.4**	72.1	68.7	-
CTransR	81.5	89.0	36.4	71.2	80.8	38.6	90.1	73.8	70.2	-
TransD	80.7	85.8	**47.1**	75.6	80.0	54.5	80.7	77.9	74.2	-
Single	66.5	88.8	39.8	79.0	66.4	51.9	85.6	81.5	76.6	66.2
Joint	**82.7**	**89.1**	45.0	**80.7**	**81.7**	**57.7**	87.4	**82.8**	**78.7**	**79.1**

Table 2: Evaluation results on entity prediction of head and tail entities (%).

scores calculated by $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2$. The relational fact (h, r, t) is expected to have smaller score than any other corrupted triples.

We follow previous works and use the proportion of correct entities in Top-10 ranked entities (Hits@10) as the evaluation metric. As mentioned in (Bordes et al., 2013), a corrupted triple may also exist in knowledge graphs, which should not be considered as incorrect. Hence, before ranking, we filter out those corrupted triples that have appeared in FB15K.

The relations in knowledge graphs can be divided into four classes: 1-to-1, 1-to-N, N-to-1 and N-to-N relations, where a "1-to-N" relation indicates a head entity may correspond to multiple tail entities in knowledge graphs, and so on. For example, the relation (*Country*, `PresidentOf`, *Person*) is a typical "1-to-N" relation, because there used to be many presidents for a country in history. We report the average Hits@10 scores when predicting missing head entities and tail entities with respect to different classes of relations. We also report the overall performance by averaging the Hits@10 scores over triples and over relations.

Since the evaluation setting is identical, we simply report the results of SE, SME, TransE, TransH, TransR/CTransR, TransD from (Bordes et al., 2011, 2012, 2013; Wang et al., 2014b; Lin et al., 2015; Ji et al., 2015). The evaluation results on entity prediction is shown in Table 2. The model for knowledge representation without joint learning in our framework is named as "Single". From Table 2 we observe that:

(1) The joint model almost achieves improvements under four classes of relations when predicting head and tail entities. This indicates the performance of joint learning is consistent and robust.

(2) The improvements on "1-to-1", "1-to-N" and "N-to-1" relations are much more significant as compared to those on "N-to-N". This indicates that our joint model is more effective to embed textual relations for those deterministic relations.

(3) Our joint model achieves improvement of more than 13% than Single when averaging over relations. This indicates that, our joint model can take advantages of plain texts and greatly improve representation power in relation-level.

(4) In FB15K, the relation numbers in different relation classes are comparable, but more than 80% triples are instances of "N-to-N" relations. Since the improvement of the joint model on "N-to-N" relations is not as remarkable as on other relation classes, hence the overall superiority of our joint model seems not so notable when averaging over triples as compared to averaging over relations.

4.3.2 Results of Relation Prediction

The task aims to predict the missing relation between two entities based on their embeddings. More specifically, we need to predict the relation when given a triple $(h, ?, t)$. In this task, for each missing relation, the system is asked to find one best result, according to similarity scores calculated by $\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|_2$. Because the number of relations is much smaller, compared with the number of entities, we use the accuracy of Top-1 ranked relations as the evaluation metric. Since some entities may have more than one relation between them, we also filter out those triples with corrupted relations appeared in knowledge graphs. We report the overall evaluation results as well as those in different relation classes.

Tasks	Relation Prediction				
Category	1-to-1	1-to-N	N-to-1	N-to-N	All
Single	24.1	83.0	80.4	92.5	87.2
Joint	**40.9**	**89.4**	**87.1**	**94.6**	**91.6**

Table 3: Evaluation results on relation prediction (%).

The evaluation results are shown in Table 3. From Table 3 we observe that, our joint model outperforms Single consistently in different classes of relations and in all. The joint model also achieves more significant improvements on “1-to-1”, “1-to-N” and “N-to-1” relations. The observations are compatible with those on entity prediction.

5 Conclusion and Future Work

In this paper, we propose a model for joint learning of text and knowledge representations. Our joint model embeds entities, relations and words in the same continuous latent space. More specifically, we adopt deep neural networks CNN with multilevel-attention to encode textual relations for joint learning of relation embeddings. In experiments, we evaluate our joint model on two tasks including three parts, entity prediction, relation prediction and relation extraction. Experiment results show that our joint model can effectively perform representation learning from both knowledge graphs and plain text, and obtain more discriminative entity and relation embeddings for prediction. In future, we will explore the following research directions:

(1) The part for knowledge representation in our paper is equivalent to TransE. Our joint model is also capable to incorporate other knowledge representation models instead of TransE, such as TransH, TransR or more. In future we will explore their capability in our joint model.

(2) We will also take more rich information in our joint model, such as relation paths in knowledge graphs, and the textual relations represented by more than one sentence in a paragraph or document. These information can also be used to incorporate into knowldege graphs.

These future work will further improve performance over knowledge and text representation, this may let the joint model make better use of knowledge and text.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. *Dbpedia: A nucleus for a web of open data*. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

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 KDD*. pages 1247–1250.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2012. Joint learning of words and meaning representations for open-text semantic parsing. In *AISTATS*. volume 351, pages 423–424.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*. pages 2787–2795.

Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. 2011. Learning structured embeddings of knowledge bases. In *Proceedings of AAAI*. pages 301–306.

Bhuwan Dhingra, Hanxiao Liu, William W Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv preprint arXiv:1606.01549* .

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 601–610.

Cıcero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of ACL-IJCNLP*. volume 1, pages 626–634.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *JMLR* 11:625–660.

Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. 2015. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, pages 623–632.

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 1693–1701.

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*. Association for Computational Linguistics, pages 541–550.

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 *Proceedings of AAAI*.

Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81(1):53–67.

Ni Lao, Tom Mitchell, and William W Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of EMNLP*. pages 529–539.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. volume 1, pages 2124–2133.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *Proceedings of ICLR* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*. pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of HLT-NAACL*. pages 746–751.

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

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *HLT-NAACL*. pages 777–782.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL-IJCNLP*. pages 1003–1011.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770* .

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

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas .

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. *arXiv preprint arXiv:1604.05525* .

Alessandro Sordoni, Phillip Bachman, and Yoshua Bengio. 2016. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245* .

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of WWW*. ACM, pages 697–706.

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*. Association for Computational Linguistics, pages 455–465.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of EMNLP*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM* 57(10):78–85.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014a. Knowledge graph and text jointly embedding. In *Proceedings of EMNLP*. pages 1591–1601.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014b. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI*. pages 1112–1119.

Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction.

Jiawei Wu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Knowledge representation via joint learning of sequential text and knowledge graphs. *arXiv preprint arXiv:1609.07075* .

Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. 2015. Transg: A generative mixture model for knowledge graph embedding. *arXiv preprint arXiv:1509.05488* .

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of EMNLP*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal*. pages 17–21.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING*. pages 2335–2344.

Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2016. Incorporating relation paths in neural relation extraction. *arXiv preprint arXiv:1609.07479* .

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006* .