An Attention-based Model for Joint Extraction of Entities and Relations with Implicit Entity Features

Yan Zhou
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
zhouyan@iie.ac.cn

Longtao Huang
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
huanglongtao@iie.ac.cn

Tao Guo
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
guotao@iie.ac.cn

Songlin Hu
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
husonglin@iie.ac.cn

Jizhong Han
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
hanjizhong@iie.ac.cn

ABSTRACT

Extracting entities and relations is critical to the understanding of massive text corpora. Recently, neural joint models have shown promising results for this task. However, the entity features are not effectively used in these joint models. In this paper, we propose an approach to utilize the implicit entity features in the joint model and show these features can facilitate the joint extraction task. Particularly, we use the hidden-layer vectors extracted from a pre-trained named entity recognition model as the entity features. Thus, our method does not need to design the entity features by hand and can benefit from the new development of named entity recognition task. In addition, we introduce an attention mechanism in our model which can select the informative parts of the input sentence to the prediction. We conduct a series of experiments on a public dataset and the results show the effectiveness of our model.

CCS CONCEPTS

• **Computing methodologies → Information extraction**.

KEYWORDS

Relation extraction; Implicit entity features; Attention mechanism; Neural network

Tao Guo is the corresponding author.

WWW '19 Companion, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6675-5/19/05.
https://doi.org/10.1145/3308560.3317704

ACM Reference Format:
Yan Zhou, Longtao Huang, Tao Guo, Songlin Hu, and Jizhong Han. 2019. An Attention-based Model for Joint Extraction of Entities and Relations with Implicit Entity Features. In *Companion Proceedings of the 2019 World Wide Web Conference (WWW '19 Companion), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3308560.3317704

1 INTRODUCTION

The extraction of entities and relations from unstructured text is an important task in information extraction and natural language processing (NLP). The goal of the task is to extract entities and their semantic relations from an unstructured input sentence. For example, in a sentence *"Donald Trump is the 45th and current president of the United States"*, *"Donald Trump"* and *"United States"* are the entities, the relation between them is *"president of country"*. The extracted information can be used for various kinds of downstream tasks such as question answering and knowledge base population. The methods for this task can be classified into two categories: pipelined models and joint models.

The pipelined models treat this problem as two separate tasks: named entity recognition (NER) [18, 19] and relation classification [22, 36]. These methods first identify the entity mentions in the sentence. And then, they take the identified mentions as correct entities and predict the relations between them. Therefore, the results of named entity recognition may have an impact on relation classification and lead to error propagation between them [14].

Different from the pipelined approaches, the joint models simultaneously identify the entity mentions and relations. These models always take an sentence as input and predict the entities and relations at the same time. Such joint methods have been proved to achieve better performance than the pipelined models. In earlier researches, the joint extraction systems typically rely on handcrafted features, however the design of these features is time-consuming. With the capability of automatic feature learning, neural network (NN) based models are proposed to resolve the relation extraction problem [12, 17, 30, 37, 38]. These joint models can utilize the inter-dependence of the entities and relations. But existing studies generally employ word embeddings as input and do not exploit external entity features in their models. And we believe that the external entity features are beneficial to the joint extraction models, since the output triplets of the task are comprised by the entities and their relations.

In this paper, we propose an attention-based joint model enhanced with implicit entity features for the extraction task. In particular, we build our model upon the tagging scheme proposed by Zheng et al. [38]. Firstly, inspired by the work of Gao et al. [6], we dump the hidden layer vectors of a pretrained named entity recognition model as the entity features. These vectors contain rich entity information about each word, yet do not require manual design. We use these vectors as the entity features of our joint model. Through the utilization of these features, our method can easily benefit from the new development and the external training data of the NER task. In addition, when the model extracts the entities and relations, there are some tokens play a more important role than others. In the example sentence above, when the model predict the tag of word "Donald", the words representing the other entity (i.e. "United" and "States") and the word indicating the relations of the entities (i.e. "president") are more important than others. To utilize these information, we propose an attention mechanism in our model.

In summary, our main contributions are as follows:

- We propose an approach to integrate implicit entity features to the joint extraction task and show that these features can facilitate this task.
- We design an attention mechanism in our model, thus our model can focus on the informative words for the prediction.
- A series of experiments conducted on a public dataset demonstrate the effectiveness of our model.

The rest of this paper is structured as follows. We describe the related work of the relation extraction task in Section 2. We introduce the preliminary knowledge in Section 3. Section 4 describes our proposed model. The experiments and results are detailed in Section 5. In Section 6, we conclude this paper.

2 RELATED WORK

The task of relation extraction is to extract triplets that are composed of two entities and the relation between these two entities. There have been many studies on the extraction of entities and relations, and the methods can be roughly divied into two categories: the pipelined methods and the joint methods. The pipelined methods first perform the named entity recognition to identify entity mentions [18, 19], and then classify the relation between two entities [22, 36]. However, the pipelined methods ignore the dependency between two subtasks, which may lead to error delivery [14]. To solve this problem, the joint methods are proposed to extract entities and relations simultaneously. Therefore, the problem we focused is related to named entity recognition, relation classification and joint extraction of entities and relations. Our method is also related to attention models.

2.1 Named Entity Recognition

Named entity recognition (NER) has a long history in the field of natural language processing. Earlier researches focused on linear statistical models, such as Hidden Markov Models (HMM) [39] and Conditional Random Fields (CRF) [33]. These models relied on hand-crafted features, gazetteers and other external resources to perform well. Recently, sereval neural models have been proposed for NER . Collobert et al. [5] used a CNN over a sequence of word embeddings with a CRF layer on top. Huang et al. [11] replaced the CNN encoder in Collobert et al. with bidirectional LSTM encoder. Lample et al. [13] and Chiu & Nichols [4] introduced neural models with additional bidirectional LSTM and CNN encoders to encode character-level features. Marek Rei et al. [20] presented an attention-based neural model to improve the performance by the character-level extentions. To reduce the amount of labeled training data in NER, Yanyao Shen et al. [25] combined deep learning with active learning.

2.2 Relation Classification

Relation classification is a common task in natural language processing. Apart from a few unsupervised clustering methods [3, 8], the majority of studies on relation classification have been supervised. Zeng et al. [34] proposed a convolutional deep neural network to extract lexical and sentence level features. And then, these features were fed into a softmax classifier to predict the relations. Zhang and Wang [35] investigated a temporal structured RNN with only words as input. Zhou et al. [40] used bidirectional LSTM with attention mechanism to capture the most important semantic information in a sentence. Wang et al. [29] proposed a novel convolutional neural network architecture for relation classification, relying on two levels of attention in order to better discern patterns in heterogeneous contexts.

2.3 Joint Extraction of Entities and Relations

There are a number of researches on the joint extraction of entities and relations. Most of the joint methods are feature-based models [14, 21], in which the design of handcrafted features is time-consuming. Recently, deep learning methods provide an effective way to reduce the manual work. For example, an relation extraction model depending on both word sequence and dependency tree structure was proposed to extract entities and relations [17]. Katiyar and Cardie [12] presented a novel recurrent neural network to extract semantic relations between entity mentions without having access to dependency trees. Though their proposed methods used a single model, the model needs to identify entities first and then extracts the relations between them. In order to better utilize the interactions between the outputs, some novel schemes that can jointly decode the entities and relations were proposed. Zheng et al. [38] introduced a new tagging scheme, based on what the task of entities and relations extraction is transformed into the sequence tagging problem. And they proposed an end-to-end model to solve

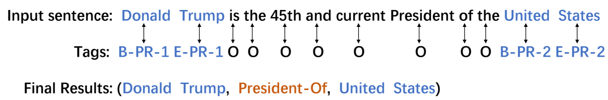

Figure 1: Gold standard annotation for the example sentence, where "PR" is short for "President-Of".

the problem. Wang et al. [30] converted the joint task into a directed graph by designing a novel graph scheme and proposed a transition-based approach to achieve joint learning through joint decoding. Though these joint models show promising results for the extraction task, they do not take advantage of external entity features.

2.4 Attention Models in NLP

Attention-based neural networks are proposed to obtain a representation weighted by the importance of tokens in the sequence. Such models have been successfully applied on many NLP applications. Bahdanau et al. [1] proposed an encoder-decoder framework with attention mechanism for machine translation. Wang et al. [31] presented an attention-based neural network for aspect-level sentiment classification. The attention mechanism can concentrate on different parts of a sentence when different aspects were taken as input. Rush et al. [23] utilized a local attention-based model that generated each word of the summary conditioned on the input sentence for sentence summarization. Yang et al. [32] introduced a hierarchical attention network for document classification. Seo et al. [24] introduced the Bi-Directional Attention Flow network for machine comprehension, which represented the context at different levels of granularity and used bidirectional attention flow mechanism to obtain a query-aware context representation.

3 PRELIMINARY KNOWLEDGE

In this section, we introduce the preliminary knowledge related to our proposed approach. We first describe the task of relation extraction in Section 3.1. Then, we briefly introduce the tagging scheme in Section 3.2. Finally, we provide a brief description of LSTM in Section 3.3.

3.1 Task Definition

The goal of relation extraction is to mine entities and their relations from unstructured texts. As shown in Figure 1, the input of the task is a sentence *"Donald Trump is the 45th and current President of the United States"* and the output of can be represented as a triplet ($Donald\ Trump, President - Of, United\ States$). The triplet is comprised of two entities and their relation. For the above example, *"Donald Trump"* and *"United States"* are the entities, and *"President-Of"* denotes the relation between the them.

3.2 Tagging Scheme

The tagging scheme is proposed by Zheng et al. [38] to extract the entities and their relations simultaneously. Based on this tagging scheme, the information between the output entities and relations can be fully exploited. Figure 1 presents an example of the tagging scheme. Tag 'O' represents that the corresponding word is independent of extracted results. The other tags which represent the elements in the triplet are named as relational tag. The relational tag is comprised of three parts: the word position in entity, the relation type and the relation role. The position of a word in entity is represented in "BIES" (Begin, Inside, End, Single) format. The relation type is obtained from a set of predefined relations. The relation role is represented by '1' or '2' which means the first or the second entity in the relation, respectively. According to the tag sequence, we can directly get the entities with its relation in the form of a triplet ($Entity_1, RelationType, Entity_2$). For the sentence presented in Figure 1, we can extract the relation triplet ($Donald\ Trump, President - Of, United\ States$). Then, based on this tagging scheme the goal of relation extraction is to predict the tag of each token.

3.3 LSTM

Recurrent neural networks (RNNs) are a family of neural networks which are capable of processing sequential input of unbounded and arbitrary length. Long Short Term Memory Networks (LSTMs) [9] are a special kind of RNNs. They take a sequence of vectors $(x_1, x_2, \ldots, x_n)$ as input and return another sequence $(h_1, h_2, \ldots, h_n)$ that represents some information about the sequence at every time step. LSTMs incorporate gating functions at each time step to allow the network to forget, remember and update contextual memory and mitigate problems like vanishing gradient. We use the following implementation:

$$i_t = \sigma(W_{xi}x_t + b_{xi} + W_{hi}h_{t-1} + b_{hi}) \quad (1)$$
$$f_t = \sigma(W_{xf}x_t + b_{xf} + W_{hf}h_{t-1} + b_{hf}) \quad (2)$$
$$g_t = tanh(W_{xg}x_t + b_{xg} + W_{hg}h_{t-1} + b_{hg}) \quad (3)$$
$$o_t = \sigma(W_{xo}x_t + b_{xo} + W_{ho}h_{t-1} + b_{ho}) \quad (4)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (5)$$
$$h_t = o_t \odot tanh(c_t) \quad (6)$$

where σ is the sigmoid function, and $\odot$ is the element-wise product.

4 METHODOLOGY

In this section, we describe the proposed approach in detail. Following recent studies, we extract the entities and relations jointly in a single model. In particular, we follow zheng et al. [38], casting the joint extraction task as a sequence tagging problem. Given an input sentence $S = (w_1, w_2, \ldots, w_n)$, the output of the model is a sequence of tags $T = (t_1, t_2, \ldots, t_n)$ corresponding to the sentence. Based on these tags, we can directly get the entities and relations. Figure 2 shows the overall architecture of our model which has four main components: an embedding layer, an encoding layer, an attention layer and a decoding layer. The embedding layer represent each input token by multi-level features. The encoding layer learns the contextual information of the input tokens. The attention layer selects the important information from the output of the encoding layer. The decoding layer predicts the tag for each word based on the vector computed by the attention layer.

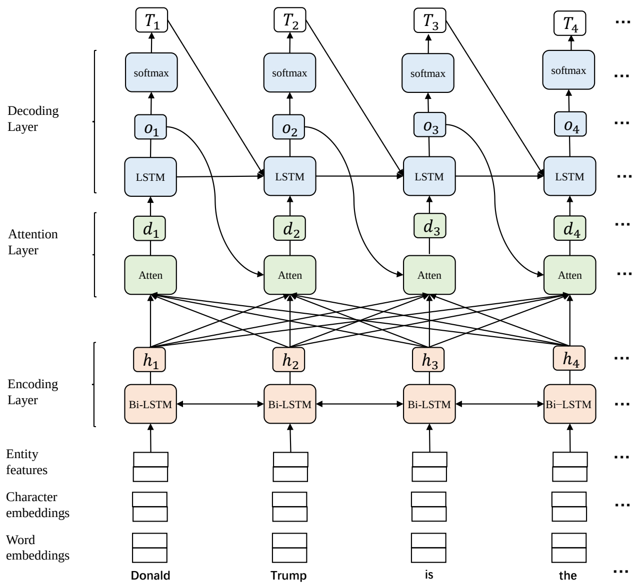

Figure 2: Our network structure for joint extraction of entities and relations.

4.1 Features

We utilize the word embeddings, character embeddings and implicit entity features to represent the input words.

Word Embeddings. For each word w_t in S, we look up the embedding matrix to get its word-level representation ww_t. It has been reported that word embeddings learned from significant amounts of unlabeled data are far more satisfactory than the randomly initialized embeddings [5]. Therefore, we initialize the matrix by word embeddings pretrained by word2vec model [15].

Character Embeddings. It has been demonstrated that character embeddings are effective to handle the out-of-vocabulary problem for several NLP tasks such as named entity recognition [20] and dependency parsing [2]. Thus, we use the character sequence of each input token to produce its character-level representation. Each word is broken up into individual characters, and these characters are mapped to a sequence of character embeddings $(c_1, c_2, \ldots, c_L)$. After that, we adopt a bidirectional LSTM to generates the character embedding wc_t for the word w_t.

Implicit Entity Features. We introduce a method to integrate implicit entity features for the joint extraction of entities and relations in this section. Intuitively, the external entity features are useful for the joint extraction task. However, the previous neural joint methods do not consider this kind of features. Since designing these features manually is always time consuming, we propose to incorprate the automatically generated entity features in our model. Inspired by the work of Gao et al. [6], we leverage the hidden vectors in a pre-trained named entity recognition model to generate the implicit entity features. We first pre-train a named entity recognition model on an existing named entity recognition dataset. And then, the input sentences are fed into this model, and the hidden vectors which contain rich entity information are dumped as the entity features. Thus, our method dose not need to manually design these features and can easily benefit from the latest developments of NER task.

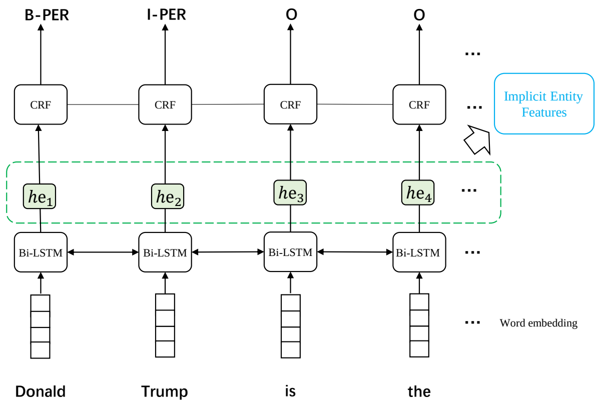

Figure 3: Entity Features Generation Model.

As shown in Figure 3, we model named entity recognition as a sequence tagging problem and employ an simplified BLSTM-CRF model as our entity features generation model. First of all, each token is represented by its word embedding. Then we feed these vectors into a bidirectional LSTM. Bidirectional LSTM contains two separate LSTMs to capture both the past and future information, both of the two information are beneficial to the named entity recognition. For the task of named entity recognition, it is important to consider the constrains between tags in neighborhood (e.g., I-PER can not follow B-LOC). And Conditional random fields (CRF) are effective to learn these constrains. Therefore, a standard CRF layer is used on the top of the model to jointly predict the tags.

To obtain the implicit entity features, we pre-train the above model on existing named entity recognition datasets first. And then, for a given sentence $(w_1, w_2, \ldots, w_n)$, we directly dump a sequence of hidden states $(he_1, he_2, \ldots, he_n)$ from the bidirectional LSTM layer of the model:

$$(he_1, he_2, \ldots, he_n) = BLSTM_{entity}(w_1, w_2, \ldots, w_n) \tag{7}$$

In this way, we get the implicit entity features representation he_t of the t-th word in the input sentence.

4.2 Encoding Layer

In the encoding layer, we adopt a bidirectional LSTM which can combine the forward and backward context of a word to encode the sentence. We use x_t to denote concatenation of ww_t, wc_t and he_t. Thus, the input of the encoding layer is a sequence of vectors $(x_1, x_2, \ldots, x_n)$. And this layer receives these vectors as input and computes the t step hidden state h_t as follows:

$$\overrightarrow{h_t} = LSTM_{encoder}(x_t, \overrightarrow{h_{t-1}}) \tag{8}$$

$$\overleftarrow{h_t} = LSTM_{encoder}(x_t, \overleftarrow{h_{t+1}}) \tag{9}$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{10}$$

Thus, we obtain the representation of each input token containing the contextual information in the encoding layer.

4.3 Attention Layer

In order to effectively utilize the information of the relevant tokens in the sentence, we design an attention layer as depicted in Figure 4. First, we propose a Tag-Aware attention for the purpose of selecting the relevant information of the word. Then, we adopt a gate to fuse this relevant semantic information and the word information obtained from the encoding layer. The details of this layer will be illustrated as follows.

Tag-Aware Attention The key components for predicting the tag of each input word may appear anywhere in the sentence. To better utilize these informative words, we employ an attention mechanism which is named as Tag-Aware Attention. The Tag-aware attention allows the model to select the relevant parts of the source sentence for the prediction of the tag.

In order to compute the relevant word representation of the t-th word, we first calculate an attention score $\alpha_{j,t}$ to the j-th word ($1 \le j \le n; j \ne t$) as:

$$e_{j,t} = h_j \cdot o_{t-1} \tag{11}$$

$$\alpha_{j,t} = \frac{exp(e_{j,t})}{\sum_{k=1}^{t-1} exp(e_{k,t}) + \sum_{k=t+1}^{n} exp(e_{k,t})} \tag{12}$$

where h_j is the hidden state of the j-th word calculated by the encoding layer, and o_{t-1} represents the output status computed by the decoding layer. After that the vector ha_t representing the informative words in the sentence is computed as the summation vector weighted by $\alpha_{j,t}$:

$$ha_t = \sum_{j=1}^{t-1} \alpha_{j,t} h_j + \sum_{j=t+1}^{n} \alpha_{j,t} h_j \tag{13}$$

Fusion Gate Based on the attention vector ha_t and the word vector obtained from the encoding layer h_t, we design a fusion gate to combine them. When predicting the tag of a word, the gate allows the model to trade off the information used from ha_t and h_t, both of which are important to the successful prediction.

Therefore, instead of just concatenating ha_t with the vector h_t, the two vectors are added together using a weighted sum. And the weights are predicted by a two-layer network. Let d_t denotes the output of the attention layer corresponding to the t-th word in the sentence, we calculate it as follows:

$$g_t = \sigma(W_g^{(3)} tanh(W_g^{(1)} ha_t + W_g^{(2)} h_t)) \tag{14}$$

$$d_t = g_t ha_t + (1 - g_t) h_t \tag{15}$$

where $W_g^{(1)}, W_g^{(2)}, W_g^{(3)}$ are weight matrices for calculating g_t, and σ is the logistic function. The vector g_t acts as the weight between ha_t and h_t, which makes the model can dynamically merge the two vectors. In addition, g_t is computed for every word in the sentence, which allows the model to be more flexibly for making different decisions for diverse words.

4.4 Decoding Layer

In the decoding layer, both the tag embeddings and the vectors computed by the attention layer are used as the input. And we adopt LSTM to learn the dependency of the tags and generate

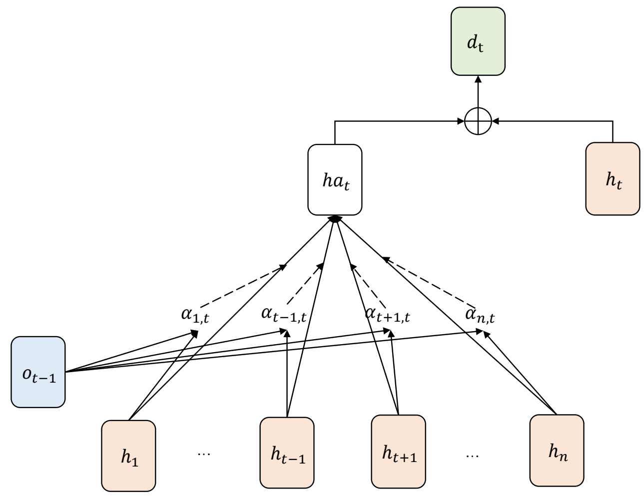

Figure 4: Overview of the Attention Mechanism.

vectors representing the output states:

$$\overrightarrow{o_t} = LSTM_{decoder}(d_t + W_T T_{t-1}, \overrightarrow{o_{t-1}}) \tag{16}$$

where o_t represents the hidden state of the t step, d_t denotes the vector from the attention layer; T_{t-1} is the tag embedding directly converted by the t-1-th tag, W_T is the weight matrix. Finally we adopt a softmax classifier to compute normalized entity tag probabilities based on the vector o_t:

$$s_t = W_s o_t + b_s \tag{17}$$

$$p(tag_t | S, tag_{t-1}) = \frac{exp\ (s_t)_{tag_t}}{\sum_{tag'_t \in TAG} exp\ (s_t)_{tag'_t}} \tag{18}$$

where W_s denotes the weight matrix, b_s is the bias term, tag_t represents the tag of the t-th word and TAG is the tag set.

4.5 Objective Function

We use the bias training goal presented in [38] to maximize the log-likelihood of the data:

$$L = max \sum_{m=1}^{|\mathbb{D}|} \sum_{t=1}^{L_m} (log(p(tag_t^m | S_m, tag_{t-1}^m)) \cdot I(O) \tag{19}$$
$$+ \alpha \cdot log(p(tag_t^m | S_m, tag_{t-1}^m)) \cdot (1 - I(O)))$$

where $|\mathbb{D}|$ is the size of training set, L_m is the length of sentence S_m, tag_t^m denotes the tag of the t-th word in sentence S_m, $I(O)$ is a switching function to distinguish the loss of tag 'O' and relational tags:

$$I(O) = \begin{cases} 1, \ if \ tag \ = \ 'O' \\ 0, \ if \ tag \ \ne \ 'O' \end{cases} \tag{20}$$

α is the bias weight which is used to control the influence of relational tags.

Table 1: Dimension sizes

Model	Network Structure	Size
Our Model	Word Embedding	300
	Char Embedding	20
	$LSTM_{char}$	25
	Entity Features	100
	$LSTM_{encoder}$	300
	$LSTM_{decoder}$	600
	Tag Embedding	50
Entity Features Generation Model	Word Embedding	300
	$LSTM_{entity}$	50

Table 2: Comparison with baselines

Method	**Prec.**	**Rec.**	**F1**
FCM	0.553	0.154	0.240
DS+logistic	0.258	0.393	0.311
LINE	0.335	0.329	0.332
MultiR	0.338	0.327	0.333
DS-Joint	0.574	0.256	0.354
CoType	0.423	0.511	0.463
LSTM-LSTM-Bias	0.615	0.414	0.495
Transition-Based	0.643	0.421	0.509
Our model	0.640	0.464	**0.538**

5 EXPERIMENTS

5.1 Experiment Setup

Dataset and Evaluation Metrics. We evaluate our proposed model with the public dataset NYT[1] which is developed by Ren et al. [21]. There are $353k$ triplets in the training data and 3880 triplets in the test set. The test set is manually annotated to guarantee its quality. The size of the relation type in the dataset is 24. In addition, we use CoNLL-2003 dataset [28] to pre-train the entity features generation model.

We adopt Precision(Prec), Recall(Rec) and F1-scores to evaluate our method. E1, E2 and R represent the first entity, the second entity and the relation type in the triplet, respectively. A triplet is regarded as correct when E1, E2 and R are all correct [21, 30, 38]. Similar to the previous work, we randomly sample 10% data from the test set as our validation set and use the remaining data as evaluation.

Hyperparameters. For experiments, we initialize our word embeddings with 300-dimensional embedding vectors trained on NYT corpus. For the tag embedding, we initialize them by random weights. We update the parameters of our model by backpropagation using RMSprop [27] with learning rate 5×10^{-4} and mini-batch size 50. For regularization, we employ dropout operation with dropout rate of 0.3 for the word embeddings. For the entity features generation model, we select RMSprop with learning rate 0.01 to train the model. The dimension of the vectors in our model and the entity features generation model are shown in Table 1. The bias term is set to 10.

5.2 Comparison with Baselines

Baselines. We compare our model against several baseline models for relation extraction, which can be divided into the following categories: the pipelined methods, the joint extraction methods and the end-to-end methods.

For the pipelined methods, the NER results are obtained by CoType [21], and then several classical relation classification methods are applied to detect the relations. The pipelined methods compared in this paper are as follows:

- **FCM** [7] a method which combines unlexicalized hand-crafted features with learned word embeddings.
- **DS+logistic** [16] combines distant supervision and syntactic parse features to improve the performance of relation extraction.
- **LINE** [26] is a network embedding method, which is suitable for arbitrary types of information networks.

We compare our model with the following joint extraction methods:

- **MultiR** [10] presents a novel approach for multi-instance learning with overlapping relations.
- **DS-Joint** [14] presents an incremental joint framework to simultaneously extract entity mentions and relations using structured perceptron with efficient beam-search.
- **CoType** [21] formulates the joint entity and relation mention typing problem as a global embedding problem.

The end-to-end methods compared with our model are as follows:

- **LSTM-LSTM-Bias** [38] converts the entities and relations extraction to a sequence tagging problem and jointly extracts them by an end-to-end model.
- **Transition-Based** [30] converts the joint task into a directed graph by designing a novel graph scheme and proposes a transition-based approach to extract the relations.

Table 2 presents the performance of our model as well as all of the baselines mentioned above. We observe that our end-to-end model achieves significant improvements over all the baselines in F1 score. In particular, it achieves 2.9% improvement over the best end-to-end methods. This demonstrates the effectiveness of our model on the extraction of entities and relations. From the above table, we can find that the joint methods perform better than the pipelined methods. This is because the joint models can avoid error delivery problem. The end-to-end models outperform the joint models. This indicates the joint decoding which can take advantage of the dependencies of the output is important for the extraction task. Our model achieves better performance compared with other end-to-end models. One reason is that employing entity features as the complementary to the standard word embeddings, our model can take advantage of extra entity information. Meanwhile, our model is able to focus on the informative parts of the input through the applying of Tag-Aware attention.

[1] https://github.com/shanzhenren/CoType/tree/master/data/source/NYT

Table 3: Ablation results

Method	Prec.	Rec.	F1
Our model	0.640	0.464	**0.538**
-Implicit Entity features	0.604	0.456	0.521
-Attention Layer	0.601	0.450	0.515
-Fusion Gate+Concat	0.599	0.469	0.526

5.3 Ablation Tests

To show the effectiveness of each components proposed in our model, we conduct a set ablation experiments and the results are presented in Table 3. The value of F1 drop 1.7%, when we do not use the implicit entity features in our model. This result shows that external entity features which are not utilized in other joint models are useful for the joint extraction task. The performance of the model will drop when the attention layer is removed. This demonstrates that the informative words in the sentence play an important role for the correct prediction. The last line presents the results, when we replace the fusion gate by merely concatenating the related word vector to the vector computed by the encoding layer. And the results prove that incorporating the relevant information of the sentence in a reasonable manner can further improve the performance.

To give a detailed analysis of the effect of the components, we show the results on different elements of the extraction triplet. E1, E2 and R represent the first entity, the second entity and the relation type in the triplet, respectively. If both the head offsets of E1 and the relation type R are correct, then the instance of (E1, R) is correct. Similarly, if both the head offsets of E2 and the relation type R are correct, then the instance of (E2, R) is correct. Regardless of relation type R, if both the head offsets of two corresponding entities are correct, the instance of (E1, E2) is correct.

Table 4, shows the results on different triplet elements. First of all, when we remove one of the component from the model, the performance of the three types triplet elements will decline. This indicates the proposed components can enhance both the entity recognition and the relation classification. Compared with the results on one of the the entities is not considered (i.e. (E1, R) and (E2, R)), the attention layer has a greater impact than the implicit entity features. This is because the informative words in the context is more important for the prediction of the relation. And from the last line of the table, we find that the effect of implicit entity features is much more than the attention layer when merely evaluating the entities of the triplet.

5.4 Impact of Implicit Entity Features Size

We make a detailed evaluation on the influence of implicit entity features size. Figure 5 shows the performance of our model when we change the dimension of the implicit entity features. We use the size of entity features in the following set $\{50, 100, 150, 200, 250\}$. From the figure we find that our model performs best when the size of entity features vector is set to 100. We can also see that the value of F1 is relatively stable when varying the dimension of implicit entity features. This demonstrates the robustness of the implicit entity features to our model.

Table 4: Ablation results on triplet elements

Elements	Method	Prec.	Rec.	F1
(E1, R)	Our Model	0.684	0.515	**0.588**
	-Implicit Entity Features	0.670	0.509	0.579
	-Attention Layer	0.669	0.501	0.573
(E2, R)	Our Model	0.663	0.499	**0.569**
	-Implicit Entity Features	0.642	0.488	0.555
	-Attention Layer	0.633	0.475	0.543
(E1, E2)	Our Model	0.667	0.501	**0.572**
	-Implicit Entity Features	0.646	0.490	0.558
	-Attention Layer	0.658	0.493	0.564

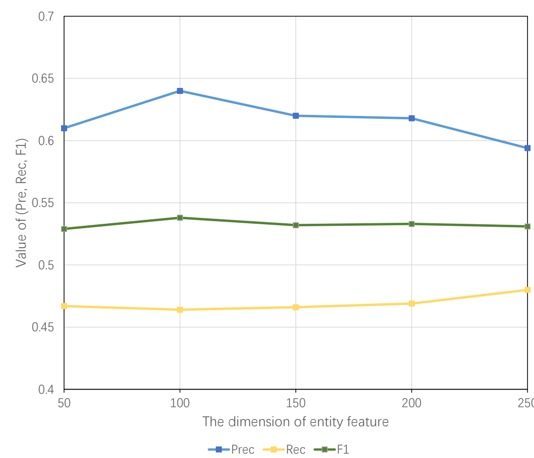

Figure 5: The results predicted by our model on different dimension of the entity feature.

5.5 Attention Visualization

We give a case study to illustrate the power of our attention mechanism. In Figure 6, we visualize the attention weights when our model predicts relational tags. We find that our model focuses more on the informative words (e.g. words stand for the entities in the triplet and words indicate the relation between the entities), when it predicts relational tags. For example, when predicting the tag of "Chad", our model pays more attention on the words "Hurley, executive, YouTube, company" which are essential for the tag "B-PC-1". Thus, our model can effectively utilize the informative words in the input sentence.

6 CONCLUSION

In this paper, we propose an attention-based model enhanced with implicit entity features for the joint extraction of entities and relations. We propose an approach to leverage the implicit entity

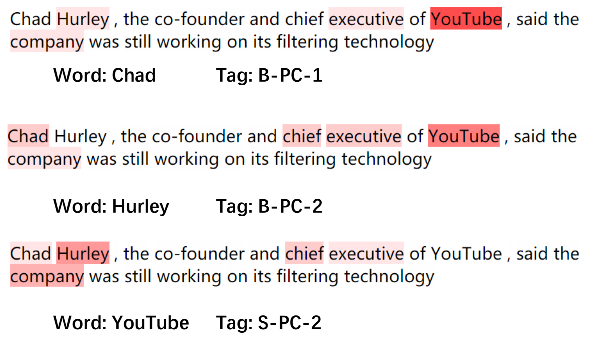

Figure 6: Visualization of attention weights when our model predicts the tags of the words {"Chad", "Hurley", "YouTube"}. We can extract the relation triplet $(Chad\ Hurley, Person-Company, YouTube)$ **from the example sentence. "PC" is short for "Person-Company" and darker color means higher weight.**

.

features in the joint extraction model. Thus, our model can take advantage of the entity features and dose not need to manually design them. In addition, we design a Tag-Aware attention mechanism which enables our model to select the informative words to the prediction. And our final model achieves competitive performance on a public dataset.

ACKNOWLEDGEMENTS

This research is supported in part by the National Key Research and Development Program of China (No. 2017YFB1010000) and the National Natural Science Foundation of China (No. 61702500).

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).
[2] Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2015. Improved Transition-based Parsing by Modeling Characters instead of Words with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 349–359.
[3] Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2005. Unsupervised feature selection for relation extraction. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.
[4] Jason PC Chiu and Eric Nichols. 2016. Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics* 4 (2016), 357–370.
[5] 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, Aug (2011), 2493–2537.
[6] Yuze Gao, Yue Zhang, and Tong Xiao. 2017. Implicit Syntactic Features for Target-dependent Sentiment Analysis. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 516–524.
[7] Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved Relation Extraction with Feature-Rich Compositional Embedding Models. In *EMNLP*. 1774–1784.
[8] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 415.
[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[10] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*. 541–550.
[11] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).
[12] Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint Extraction of Entity Mentions and Relations without Dependency Trees. In *ACL*. 917–928.
[13] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of NAACL-HLT*. 260–270.
[14] Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 402–412.
[15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[16] 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*. 1003–1011.
[17] Makoto Miwa and Mohit Bansal. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In *ACL*. 1105–1116.
[18] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes* 30, 1 (2007), 3–26.
[19] Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 147–155.
[20] Marek Rei, Gamal Crichton, and Sampo Pyysalo. 2016. Attending to Characters in Neural Sequence Labeling Models. In *COLING*. 309–318.
[21] Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. CoType: Joint extraction of typed entities and relations with knowledge bases. In *WWW*. 1015–1024.
[22] Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 256–259.
[23] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 379–389.
[24] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional Attention Flow for Machine Comprehension. In *ICLR*.
[25] Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep Active Learning for Named Entity Recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. 252–256.
[26] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.
[27] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4, 2 (2012), 26–31.
[28] Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 142–147.
[29] Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. (2016).
[30] Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint Extraction of Entities and Relations Based on a Novel Graph Scheme. In *IJCAI*.
[31] Yequan Wang, Minlie Huang, Li Zhao, et al. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*. 606–615.
[32] 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*. 1480–1489.
[33] Lin Yao, Chengjie Sun, Shaofeng Li, Xiaolong Wang, and Xuan Wang. 2009. CRF-based active learning for Chinese named entity recognition. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. IEEE, 1557–1561.
[34] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2335–2344.
[35] Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006* (2015).
[36] Zhu Zhang. 2004. Weakly-supervised relation classification for information extraction. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 581–588.
[37] Suncong Zheng, Yuexing Hao, Dongyuan Lu, Hongyun Bao, Jiaming Xu, Hongwei Hao, and Bo Xu. 2017. Joint entity and relation extraction based on a hybrid neural network. *Neurocomputing* 257 (2017), 59–66.
[38] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. In *ACL*. 1227–1236.

[39] GuoDong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. Association for Computational Linguistics, 473–480.
[40] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 207–212.