# Joint Extraction of Entities and Relations with a Hierarchical Multi-task Tagging Model

**Zhepei Wei**[1], **Yantao Jia**[2], **Yuan Tian**[1],
**Mohammad Javad Hosseini**[3], **Mark Steedman**[3], **Yi Chang**[1]
[1]School of Artificial Intelligence, Jilin University, Changchun, China
[2]Huawei Technologies Co., Ltd, Beijing, China
[3]School of Informatics, University of Edinburgh, Edinburgh, Scotland
zhepei.wei@gmail.com, jamaths.h@163.com, yuantian@jlu.edu.cn,
{s1583634, steedman}@inf.ed.ac.uk, yichang@jlu.edu.cn



## Abstract

Entity extraction and relation extraction are two indispensable building blocks for knowledge graph construction. Recent works on entity and relation extraction have shown the superiority of solving the two problems in a joint manner, where entities and relations are extracted simultaneously to form relational triples in a knowledge graph. However, existing methods ignore the hierarchical semantic interdependency between entity extraction (EE) and joint extraction (JE), which leaves much to be desired in real applications. In this work, we propose a hierarchical multi-task tagging model, called HMT, which captures such interdependency and achieves better performance for joint extraction of entities and relations. Specifically, the EE task is organized at the bottom layer and JE task at the top layer in a hierarchical structure. Furthermore, the learned semantic representation at the lower level can be shared by the upper level via multi-task learning. Experimental results demonstrate the effectiveness of the proposed model for joint extraction in comparison with the state-of-the-art methods.

## 1 Introduction

Entity extraction and relation extraction are crucial for building a large-scale knowledge graph (KG) with entities of different types as nodes and relations among them as edges. The relational facts in KGs are mostly stored in the form of triple $(h, r, t)$, in which $h$ and $t$ are the head and tail entities respectively, and $r$ is the relation between them. Typical KGs include Google Knowledge Graph (Singhal, 2012), Knowledge Vault (Dong et al., 2014), YAGO (Suchanek et al., 2007), DBpedia (Auer et al., 2007), Freebase (Bollacker et al., 2008), etc.

Previous works can be divided into two categories, namely, pipelined methods and joint methods. Pipelined methods like FCM (Gormley et al., 2015) solve the problem in two steps: first recognizing the entities and then classifying the relations between extracted entities. Intuitive and flexible though they may be, these methods neglect the interaction between the two steps. Additionally, they inevitably suffer from error propagation problems (Li and Ji, 2014). That is to say, the result of entity extraction may introduce noise to the next step, and further affects the extracted relations. To address these issues, joint methods have been investigated and shown the effectiveness of extracting entities and relations simultaneously through an integrated model. In other words, joint methods combine the two steps into a single joint extraction (JE) task. Among them are feature-based methods (Li and Ji, 2014; Miwa and Sasaki, 2014; Ren et al., 2017) and neural network-based methods (Zheng et al., 2017; Zeng et al., 2018; Takanobu et al., 2019). However, feature-based methods need complicated feature engineering, which is time-consuming and labor-intensive. They heavily rely on external natural language processing tools for preliminary feature extraction, leading to similarly erroneous delivery to the pipelined methods. Recent works employ neural network-based models to ease this issue and gain considerable improvement in joint extraction of entities and relations.

However, most existing methods ignore the interdependency between entity extraction (EE) and joint extraction (JE) tasks, in which the former extracts entities and assign each entity a tag indicating its type while the latter directly extracts relational triples but without identifying the entity type. In this paper, we propose to better perform JE while solving the auxiliary EE task by utilizing the interdependency between them. The extracted entity types help to identify the head entity and tail entity with respect to a relational triple to

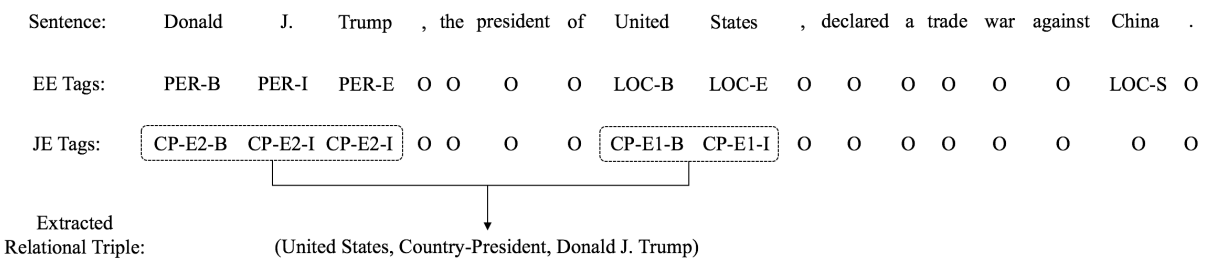


Figure 1: An example of EE and JE tags. Based on the JE tags, we can obtain the extracted relational triples of the input sentence. In this example, "CP" is the abbreviation for relation type "Country–President". Note that the JE task does not depend on the predicted tags of EE task, this figure is just for illustrating the interdependency between two tasks.

be extracted in the JE task. In addition, the partial overlap between EE and JE (they both recognize entity boundaries) can further improve both tasks via multi-task learning. Figure 1 shows an example sentence and its tags assigned by the two tasks. For a relational triple with relation type "Country-President", i.e., *(head entity, Country-President, tail entity)*. It is supposed to take an entity with type "LOC" as the head entity, and take an entity with type "PER" as the tail one according to the intrinsic semantic constraints of its relation type. Inspired by such principle, we believe that identifying the entity type "PER" of "Donald J. Trump" and entity type "LOC" of "United States" in EE will be of great help to obtain the relational triple *(United States, Country-President, Donald J. Trump)* in JE, as shown in Figure 1. To conclude in brief, the interdependency between EE and JE tasks mainly lie on two-folds: (1) Both tasks share a partial overlap, and JE is a semantically higher-level task to EE task. (2) To some extent, EE task will compensate for JE task due to JE's lack of identifying entity types.

To model the interdependency and further improve the performance of JE, we propose a hierarchical multi-task tagging (HMT) model to jointly extract entities together with relations (i.e., relational triples) from texts. The utility of multi-task learning (Caruana, 1993) owes to its capability of introducing inductive bias between tasks (Ruder, 2017), which has been exploited to improve the model's performance in related area (e.g., relation extraction (Jiang, 2009)). Specifically, in this work the EE task is organized at the bottom layer and JE task at the top layer in a hierarchical structure. Furthermore, the learned semantic representation at the lower level can be shared by the upper level via multi-task learning. Each task is a tagging module containing a bidirectional LSTM (Bi-LSTM) encoder followed by a LSTM decoder. The encoded vector in EE task is concatenated with the embedding vector of the input sentence, and then taken as the input of the encoder in JE task. Such hierarchical parameter sharing is essentially different from traditional hard/soft parameter sharing (Ruder, 2017) that ignores the hierarchical interdependency between tasks. Each task has its own module with independent parameters, but meanwhile, there are partial overlap between two tasks. Besides, we propose a multi-task objective function to train two tasks jointly, making the shared parameters better optimized to convey the learned semantic representation from EE to JE task. In this manner, our proposed model is able to utilize the interdependency between EE and JE. What's more, the model is also free from error propagation problems because the JE does not rely on the predicted tags of EE but on the learned semantic representation produced by EE encoder.

The main contribution of this work are as follows:

1. To the best of our knowledge, we are the first to investigate the interdependency between entity extraction and joint extraction tasks.

2. We propose a novel hierarchical multi-task tagging model to jointly extract entities and relations. In addition, we validate the model's capability of capturing such interdependency by an ablation test.

3. Our model substantially outperforms the state-of-the-art method on two widely used public datasets NYT10 and NYT11 with 1.6% and 6.1% improvements in terms of F1-score.

## 2 Related Work

Multi-task learning (MTL) (Caruana, 1997) is a promising framework for improving general performance, in which multiple related tasks are learned simultaneously in order to achieve better performance for each individual task. Jiang (2009)'s work is among the first to indicate that MTL can be naturally used for transferring linguistic knowledge (e.g., semantic representation and syntactic structure) since it models the commonality among tasks, and the proposed MTL model achieves better performance in relation extraction by treating classification of different relation types as related task. Similar to language model pretraining (Radford et al., 2019), MTL has been proven effective in learning general semantic representation and transferring shared knowledge among linguistic tasks (Liu et al., 2019). To obtain further improvement, Rei (2017) combines the two methods by introducing language modeling task as a related task to some sequence labeling tasks, such as entity extraction, chunking and POS tagging. Additionally, Ruder (2017) also overviews the appliance of MTL in deep neural networks and indicates that by sharing a common hidden layer representation among related tasks, the inductive bias between them can be obtained through multi-task learning for improving the performance. However, in previous works MTL is typically done with either hard or soft parameter sharing of hidden layers, which ignores the hierarchical interdependency between related tasks.

Recently, hierarchical multi-task learning has been explored with deep neural networks and shown the effectiveness in transfer learning for sequence labeling problems (Yang et al., 2017). Among the works on investigating the hierarchical semantic interdependency between linguistic tasks, perhaps the most related work to ours is (Sanh et al., 2019), which utilizes the semantic hierarchy among related tasks to better perform embedding learning. Moreover, it also suggests a hierarchy for several semantic tasks, i.e., entity extraction $<$ entity mention detection $<$ coreference resolution $=$ relation extraction. Although similar hierarchy between EE and JE seems intuitively plausible, it has not been investigated in the task of joint extraction of entities and relations with hierarchical MTL. Our work confirms this intuition and shows such hierarchical interdependency helps to better perform JE.

Previous works on entity extraction and relation extraction can be categorized into pipelined methods and joint methods. The pipelined methods (Mintz et al., 2009; Gormley et al., 2015; Tang et al., 2015) solve the problem by two steps: they first recognize entities in the sentence and then classify the relations between the extracted entities. This separated setting neglects the relevance of the two steps and causes error propagation problems since the second step is unavoidably affected by the errors introduced by the first step. To resolve the problem, many joint methods have been proposed for extracting entities and relations simultaneously. Early works (Li and Ji, 2014; Miwa and Sasaki, 2014; Ren et al., 2017) employ feature-based methods, which heavily depend on feature engineering and require much manual efforts. To reduce manual work, recent studies have investigated neural network-based methods for joint extraction. Miwa and Bansal (2016) propose a LSTM-based neural network model to jointly extract entities and relations through parameter sharing. Zheng et al. (2017) introduce a novel tagging scheme for joint extraction and convert this task into an end-to-end sequence tagging problem. Inspired by (Zheng et al., 2017), our JE task employs a similar tagging scheme. Wang et al. (2018) transform the joint extraction task into a graph problem by designing a novel graph scheme and propose a neural transition-based parsing framework to generate directed graph incrementally. Zeng et al. (2018) adopt a sequence-to-sequence (Seq2Seq) model with copy mechanism to extract multiple relational triples. Takanobu et al. (2019) apply a reinforcement learning framework to achieve joint extraction by regarding the related entities as the arguments of a relation.

However, most of these methods amount to combining the two separated steps as described in pipelined methods into a single joint extraction task. Despite their success, none of them consider the interdependency between entity extraction and joint extraction. In this work, we propose a hierarchical multi-task tagging (HMT) method to model such interdependency and jointly extract entities and relations.

## 3 Hierarchical Multi-task Tagging Model

In this section, we elaborate the design of two tagging tasks in a hierarchical setting, namely entity extraction (EE) task and joint extraction (JE)

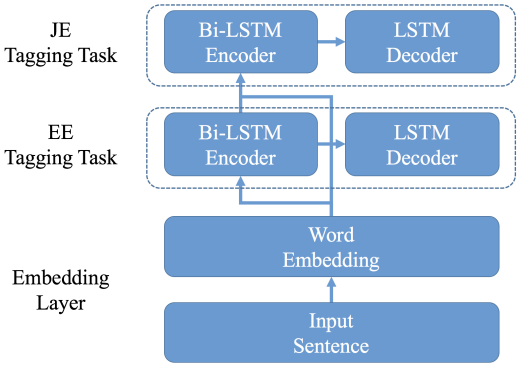


Figure 2: The structure of the proposed HMT model. Each task is a tagging module consisting of a Bi-LSTM encoder and a LSTM decoder.

task. Given the input sentence, the EE task recognizes the entity mentions as well as their types while the JE task recognizes the entity mentions as well as their involved relations but without identifying the entity types. In this setting, the proposed HMT model is able to capture the interdependency between EE and JE via multi-task learning and jointly extract entities and relations in an end-to-end fashion. Figure 2 shows the structure of the proposed model.

More precisely, an input natural language sentence is firstly converted into dense vector representation, namely word embeddings, by embedding layer and then fed into higher levels, i.e., EE task and JE task. Each task is a tagging module with the same encoder-decoder structure, in which a bidirectional *long short-term memory* (Bi-LSTM) layer is used as encoder and a LSTM layer is used as decoder. To capture the interdependency between the two tasks, the encoded vector of EE task is concatenated with input word embeddings and then fed into the encoder of JE task.

Before going deep inside the neural network, we first introduce the tagging schemes of the two tasks. There are two popular tagging schemes in the realm of sequence tagging tasks, namely "BIOES" (Begin, Inside, Outside, End, Single) scheme and "BIO" scheme. Previous works (Dai et al., 2015; Yang et al., 2018) have proved that models using "BIOES" are better than those using "BIO" in most cases. In this work, for EE task, we also apply the widely used "BIOES" scheme to assign a unique tag for each word. Specifically, tag "O" means the corresponding word is irrelevant to the extracted result and other tags except "O" are composed of two parts: *entity type* and *word position*. The *entity type* includes person (PER), location (LOC), organization (ORG) and miscellaneous (MISC). The *word position* contains "BIES", where "BIE" represents the position of a word in an entity as begin, inside and end, and "S" represents the word is a singleton entity (an entity mention with only one word).

For the JE task, each tag is composed of three parts: *relation type*, *relation role* and *word position* with respect to the relational triple form of $(h, r, t)$. Different from the EE task, we use the "BIO" scheme due to the observation (from our empirical experiments) that increasing the number of candidate tags will hurt the performance of JE task. The *relation type* is predefined in the corpus and *relation role* is either head entity (E1) or tail entity (E2). The *word position* part is similar to that in EE task. Figure 1 presents an example illustrating the two tagging schemes. Here we take the JE tags for example in detail and show how we obtain the relational triples based on the tags. "Donald" is the first word of entity "Donald J. Trump", which is involved in the relation "Country–President" as the tail entity. Hence, the tag of "Donald" is "CP-E2-B". Meanwhile, the first word "United" in "United States" is tagged as "CP-E1-B" since "United States" is involved in the relation "Country–President" as the head entity. To obtain the final relational triples, we combine the entities with the same relation type according to their tags. Now we know "Donald J. Trump" and "United States" share the same relation type "Country–President" as well as their respective role in the relation. So the final extracted result of JE task is (United States, Country–President, Donald J. Trump).

### 3.1 Embedding Layer

Word embedding is proposed (Hinton, 1986) to convert the words into vector representations to capture the syntactic and semantic meanings of words. An input sentence $s = \{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_n\}$, where $\mathrm{x}_i$ is the i-th word, can be represented by a sequence of vectors $\{w_1, w_2, ..., w_n\}$ and then fed into neural networks.

We use an embedding layer to transform the 1-hot represented words to dense vectors with the pre-trained Glove (Pennington et al., 2014) word embedding weights[1]. The word embeddings are

[1] https://nlp.stanford.edu/projects/glove/

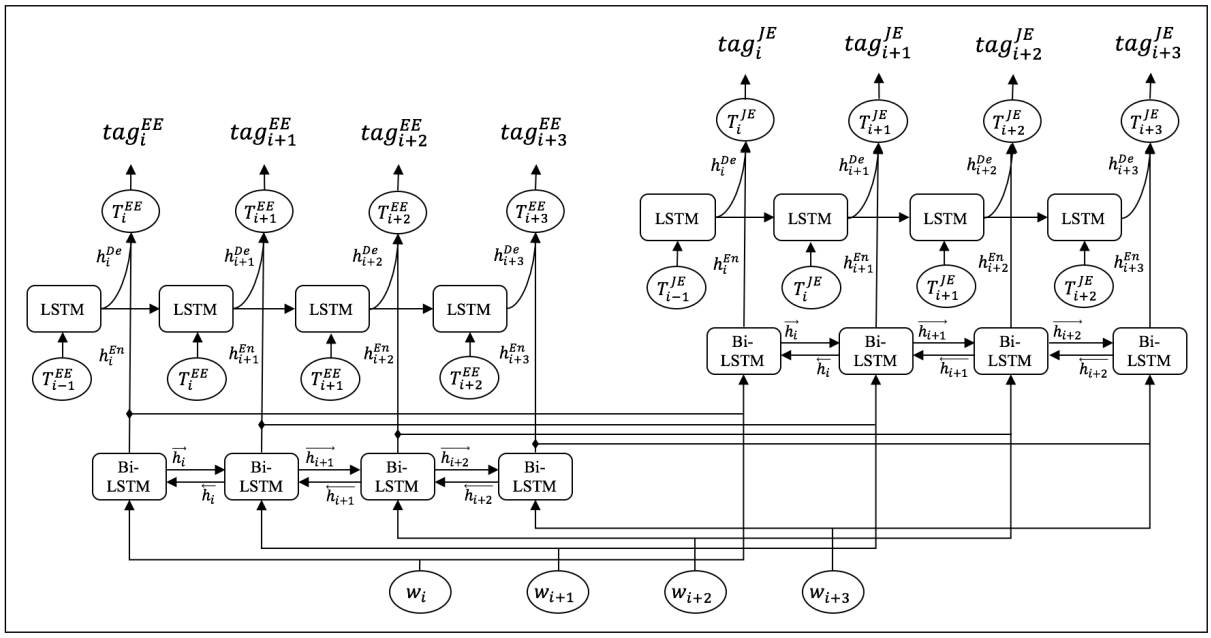


Figure 3: The Hierarchical Multi-task Tagging Model. The left part represents the EE task and the right one is JE task. Both of them hold the same encoder-decoder structure, and they are organized in a hierarchical setting as shown above. "En" represents encoder and "De" represents decoder.

fine-tuned during training.

### 3.2 Encoder-Decoder Module

As shown in Figure 2, our EE and JE tasks adopt the same encoder-decoder structure. The minor difference between them is the input of their encoders, which is described in the following sections.

#### 3.2.1 Bi-LSTM Encoder

Lample et al. (2016) have proved bidirectional *long short-term memory* (Bi-LSTM) to be effective for capturing semantic information for each word in the input sentence by processing the sequence in both directions with two parallel LSTM layers. In this work, we use a popular LSTM variant introduced by (Gers and Schmidhuber, 2000). The detailed operations are as follows:

$$
\begin{aligned}
f_t &= \sigma(W_{f_x} x_t + W_{f_h} h_{t-1} + W_{f_c} c_{t-1} + b_f) \\
i_t &= \sigma(W_{i_x} x_t + W_{i_h} h_{t-1} + W_{i_c} c_{t-1} + b_i) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{c_x} x_t + W_{c_h} h_{t-1} + b_c) \\
o_t &= \sigma(W_{o_x} x_t + W_{o_h} h_{t-1} + W_{o_c} c_t + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned} \tag{1}
$$

where at time step $t$ $(1 \leq t \leq n)$, $x_t, h_t$ and $c_t$ are input vector, hidden state vector and cell state vector respectively, $f_t, i_t$ and $o_t$ are forget gate, input gate and output gate respectively, and $b_{(.)}$ is the bias weight. For each input $x_t$ in a sequence of length $n$, the forward LSTM encodes $x_t$ by considering the contextual information from $x_1$ to $x_t$, and the encoded vector is denoted as $\overrightarrow{h_t}$. Similarly, the backward LSTM encodes $x_t$ based on the contextual information from $x_n$ to $x_t$, and the encoded vector is denoted as $\overleftarrow{h_t}$. We then concatenate $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ to represent the t-th word of the input sequence and denote it as $h_t^{En} = [\overrightarrow{h_t}, \overleftarrow{h_t}]$.

As for the difference between the encoders in EE and JE tasks, the input of the EE encoder is the word embedding vector $w_t$ of t-th word $\mathrm{x}_t$, i.e., $x_t = w_t$. For JE task, the input of the encoder is the concatenation of embedding vector $w_t$ and encoded vector $h_t^{En}$ from EE task as shown in Figure 3.

#### 3.2.2 LSTM Decoder

We also design a novel LSTM decoder to predict the tag sequences. To explicitly model the interactions between tags, we fuse the tag information into the input of LSTM at each time step $t$. The detailed operations are as follows:

$$
\begin{aligned}
f_t &= \sigma(W_{f_T} T_{t-1} + W_{f_h} h_{t-1} + b_f) \\
i_t &= \sigma(W_{i_T} T_{t-1} + W_{i_h} h_{t-1} + b_i) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{c_T} T_{t-1} + W_{c_h} h_{t-1} + b_c) \\
o_t &= \sigma(W_{o_T} T_{t-1} + W_{o_h} h_{t-1} + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned} \tag{2}
$$

where $T_t$ is the predicted tag vector of the t-th word $x_t$. The decoder hidden state vector at time step $t$ is denoted as $h_t^{De}$, which is concatenated with encoded vector $h_t^{En}$ for computing $T_t$:

$$T_t = W_T[h_t^{En}, h_t^{De}] + b_T \tag{3}$$

We can then compute the tag probability for the t-th word $x_t$ in sentence $s_j$ based on the predicted tag vector $T_t$:

$$y_t = W_y T_t + b_y \tag{4}$$

$$p(y_t^i|s_j, \theta) = \frac{\exp(y_t^i)}{\sum_{k=1}^{N_t} \exp(y_t^k)} \tag{5}$$

where $p(y_t^i|s_j, \theta)$ is the probability of assigning the i-th tag to the t-th word $x_t$ in sentence $s_j$, $\theta$ is the model parameter and $N_t$ is the number of tags.

Note that the proposed decoder is non-trivial, since the tag interactions are vital for sequence labeling (Vaswani et al., 2016). Different from the traditional decoder (e.g., vanilla LSTM or CRF) that only takes encoded vector as input and ignores the tag interactions, our decoder can model such interactions by taking the former tag information into consideration as well.

### 3.3 Multi-task Objective Function

In our work, both EE and JE tasks are treated as sequence tagging problems with an identical structure. The two tasks hold the same form of objective function:

$$J(\theta)_{(.)} = \max \sum_{j=1}^{|D|} \sum_{t=1}^{L_j} (\log p(y_t^{G_t}|s_j, \theta)) \tag{6}$$

where $|D|$ is the size of training set, $L_j$ is the length of sentence $s_j$, $G_t$ is the gold standard tag of t-th word $x_t$ in $s_j$, and $p(y_t^{G_t}|s_j, \theta)$ is the probability of assigning $G_t$ to $x_t$, which is defined in Equation 5.

The final multi-task objective function is defined as follows:

$$J(\theta) = J_{EE}(\theta) + J_{JE}(\theta) \tag{7}$$

We train the model by maximizing the log likelihood $J(\theta)$ through stochastic gradient descent over shuffled mini-batches and the optimization algorithm is RMSprop proposed by (Tieleman and Hinton, 2012).

## 4 Experiments

### 4.1 Experimental Setting

**Datasets and Evaluation Metrics** We use the public New York Times corpus[2] to evaluate our proposed model. The corpus has two versions, which are named as NYT10 and NYT11 by Takanobu et al. (2019). For NYT10, both the training set and test set are produced by distant supervision (Riedel et al., 2010). For NYT11, the training set is also produced by distant supervision while the test set is manually annotated (Hoffmann et al., 2011). Previous works (Ren et al., 2017; Zeng et al., 2018; Takanobu et al., 2019) employed different ways of data preprocessing and the performance of these models usually differs with various customized datasets. Based on the fact that only Takanobu et al. (2019) provide a comprehensive comparison of recent works on both NYT10 and NYT11 in their paper, for fair comparison, we directly conduct experiments based on the preprocessed datasets released by (Takanobu et al., 2019), where NYT10 contains 70,339 sentences for training and 4,006 sentences for test and NYT11 contains 62,648 sentences for training and 369 sentences for test. Besides, the number of relation types is 29 in NYT10 and 12 in NYT11. We also create a validation set by randomly sampling 0.5% data from the training set for each dataset as (Takanobu et al., 2019) suggested.

Following previous works (Zheng et al., 2017; Zeng et al., 2018; Takanobu et al., 2019), we use the standard Precision, Recall, and F1-score to evaluate our model. An extracted relational triple is regarded as correct only if the head entity, tail entity and the relation type are all correct.

**Hyper-parameter Setting** We determine the hyper-parameters through grid search. For embedding layer, the dimension of word embeddings is set to be 300. For encoder-decoder module, the hidden size of LSTM cell is 300 in both Bi-LSTM encoder and LSTM decoder. The dimension of tag vector $T$ in Equation 3 is 300. We also adopt dropout and mini-batch mechanism in our model, the dropout rate is 0.5 and the batch size is 32.

[2]NYT corpus is a popular benchmark, which has been employed by many previous works, including CoType (Ren et al., 2017). In Ren's paper, there are another two datasets, i.e., Wiki-KBP and BioInfer. However, we only conduct experiments on NYT dataset because the open literature (Zheng et al., 2017) points out that the employment of such two datasets is beyond the scope of tagging-based methods.

| Method | NYT10 | | | NYT11 | | |
|---|---|---|---|---|---|---|
| | *Prec.* | *Rec.* | *F1* | *Prec.* | *Rec.* | *F1* |
| FCM (Gormley et al., 2015) | – | – | – | 0.432 | 0.294 | 0.350 |
| MultiR (Hoffmann et al., 2011) | – | – | – | 0.328 | 0.306 | 0.317 |
| CoType (Ren et al., 2017) | – | – | – | 0.486 | 0.386 | 0.430 |
| SPTree (Miwa and Bansal, 2016) | 0.492 | 0.557 | 0.522 | 0.522 | **0.541** | 0.531 |
| Tagging (Zheng et al., 2017) | 0.593 | 0.381 | 0.464 | 0.469 | 0.489 | 0.479 |
| CopyR (Zeng et al., 2018) | 0.569 | 0.452 | 0.504 | 0.347 | 0.534 | 0.421 |
| HRL (Takanobu et al., 2019) | **0.714** | 0.586 | 0.644 | 0.538 | 0.538 | 0.538 |
| HMT | 0.702 $\pm$ 0.008 | **0.623** $\pm$ **0.005** | **0.660** $\pm$ **0.003** | **0.676** $\pm$ **0.009** | 0.538 $\pm$ 0.008 | **0.599** $\pm$ **0.006** |

Table 1: Results of different methods on extraction of both entities and their relations. We run the model for 10 times and report the average results as well as the standard deviation.

### 4.2 Experimental Result

**Baselines** We compare our HMT model with several state-of-the-art extraction models, including a pipelined model (FCM) and some joint models which can be further categorized into feature-based models (MultiR, CoType) and neural network-based models (SPTree, Tagging, CopyR and HRL).

**FCM** (Gormley et al., 2015) is a compositional model that combines lexicalized linguistic contexts and word embeddings for relation extraction.

**MultiR** (Hoffmann et al., 2011) is a typical distant supervision method performing sentence-level and corpus-level extraction based on multi-instance learning algorithms to deal with noisy labels in training data.

**CoType** (Ren et al., 2017) is a domain-independent framework that jointly embeds entity mentions, relation mentions, text features, and type labels into vectors, and treats the extraction task as a global embedding problem.

**SPTree** (Miwa and Bansal, 2016) is an end-to-end relation extraction model that represents both word sequence and dependency tree structures based on bidirectional sequential and tree-structured LSTM-RNNs.

**Tagging** (Zheng et al., 2017) is an approach that treats joint extraction as a sequence tagging problem using a special tagging scheme where each tag indicates entity mention and relation type simultaneously.

**CopyR** (Zeng et al., 2018) is a Seq2Seq learning framework with a copy mechanism for joint extraction, in which multiple decoders are applied to generate relational triples.

**HRL** (Takanobu et al., 2019) is a hierarchical reinforcement learning (RL) framework that performs relation detection and entity extraction in two-level RL polices.

**Results** Table 1 shows the results. Our HMT model substantially outperforms all the baselines in terms of F1-score and achieves 1.6% and 6.1% improvements over the best neural network-based joint extraction method HRL (Takanobu et al., 2019) on NYT10 and NYT11 respectively. This demonstrates the effectiveness of our proposed model on jointly extracting entities and relations.

Note that the test set of NYT10 is produced by distant supervision and contains noisy data while the test set of NYT11 is manually annotated and can be considered as clean data. Though all the models are trained on noisy data, some principles still can be observed from the performances on the two different kinds of test set. Results on NYT10 show the proposed HMT model surpasses all the baselines especially in the aspect of recall, demonstrating the robustness of our model to noisy data. Nonetheless, the precision of our model on NYT10 is lower than that of HRL, which owes to their employment of RL for denoising the training data. Results on NYT11 also show that most neural network-based methods (SPTree, Tagging, CopyR and HRL) can obtain better performance than pipelined (FCM) and feature-based (MultiR and CoType) methods, indicating the strong power of neural networks and the superiority of extracting relational triples in a joint manner. Though SPTree gains slightly higher score in recall than our model, it needs more linguistic resources (e.g., POS tags, chunks and syntactic parsing trees). Noticeably, the precision of our HMT model is much higher than all the other models, we owe it to the multi-task learning strategy and the hierarchical setting of our method.

| | | |
|---|---|---|
| Example1 | Ground Truth: | At the center of this manufactured maelstrom is the preternaturally beauteous figure of **[Shilpa Shetty]**$_{PN-E1}$ , 31 , a Bollywood movie star from **[India]**$_{PN-E2}$ whose treatment by British contestants in the so-called reality show on television here has provoked more than 16,000 viewers to complain to regulators that she is the victim of racist bullying . |
| | Predicted: | At the center of this manufactured maelstrom is the preternaturally beauteous figure of **[Shilpa Shetty]**$_{PN-E1}$ , 31 , a Bollywood movie star from **[India]**$_{PN-E2}$ whose treatment by British contestants in the so-called reality show on television here has provoked more than 16,000 viewers to complain to regulators that she is the victim of racist bullying . |
| Example2 | Ground Truth: | Homage to **[Cambodia]**$_{AC-E2}$ was performed at Chaktomuk Conference Hall in **[Phnom Penh]**$_{AC-E1}$ on Oct. 21 , attended by the king . |
| | Predicted: | Homage to **[Cambodia]**$_{CA-E1}$ was performed at Chaktomuk Conference Hall in **[Phnom Penh]**$_{CA-E2}$ on Oct. 21 , attended by the king . |

Table 2: Case study. "Ground Truth" represents the gold standard tags of the given sentence and "Predicted" represents the output of our HMT model. "PN" is the abbreviation for "Person–Nationality". "AC" is the abbreviation for "Administrative_division–Country" and "CA" is the abbreviation for "Country–Administrative_division". Note that the relation is directed, "AC" and "CA" are actually equivalent.

| Method | NYT10 | | | NYT11 | | |
|---|---|---|---|---|---|---|
| | *Prec.* | *Rec.* | *F1* | *Prec.* | *Rec.* | *F1* |
| W/o EE Task | 0.693 | 0.611 | 0.649 | 0.652 | 0.508 | 0.571 |
| **HMT** | **0.702** | **0.623** | **0.660** | **0.676** | **0.538** | **0.599** |

Table 3: Ablation test on NYT10 and NYT11. We run the model for 10 times and report the average results.

**Ablation Test** To validate our argument about the effectiveness of our hierarchical multi-task tagging model, we further conduct a set of ablation experiments. We remove the EE task module from our HMT model to test its performance. To make it comparable, we keep all the hype-parameters unchanged and retrain the rest part from scratch. Table 3 shows the advantage of introducing EE task. After removing EE task, the scores of both precision and recall decrease significantly on both NYT10 and NYT11 datasets[3], indicating the importance of introducing EE task in our model. These observations demonstrate that the learned knowledge in EE task is successfully shared and is indeed of benefit to the JE task. It also validates the capability of our model in capturing the interdependency between the two tasks.

**Case Study** We further study the performance of HMT model by describing some cases. Table 2 presents two examples demonstrating the pros and cons of our proposed method. The first example shows that our model can exactly extract relational triple from a very long sentence. The second example reveals the flaw of our model that it fails to cope with the directed relation type. As shown in table 2, "CA" and "AC" actually express the same relation between two entities but with the opposite direction. However, our model does not perform well with directed relational triples as compared to the gold standard. This is a common case in real life and has rarely been reported as well resolved in previous works. We leave the identification of directed relations for future work.

[3]Nonetheless, the performance on NYT11 is still better than most baselines. We owe it to the novel LSTM decoder and the efficient "BIO" tagging scheme of JE.

## 5 Conclusion

In this paper, we propose a hierarchical multi-task tagging (HMT) model to jointly extract entities and relations. Compared to existing methods, our method is the first to consider the interdependency between entity extraction task and joint extraction of entities and relations task by connecting the two tasks via multi-task learning. Experimental results show that our model substantially outperforms state-of-the-art baselines on the standard New York Times (NYT) benchmark. In the future, we shall investigate some strategies to capture the directional information for better extracting the directed relational triples.

## References


Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, pages 722–735.

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.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Richard Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Hong-Jie Dai, Po-Ting Lai, Yung-Chun Chang, and Richard Tzong-Han Tsai. 2015. Enhancing of chemical compound and drug name recognition using representative tag scheme and fine-grained tokenization. *Journal of cheminformatics*, 7(S1):S14.

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*, pages 601–610.

Felix A Gers and Jürgen Schmidhuber. 2000. Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, volume 3, pages 189–194.

Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1774–1784.

Geoffrey E Hinton. 1986. Learning distributed representations of concepts. In *Proceedings of the 8th Annual Conference of the Cognitive Science Society*, volume 1, page 12.

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.

Jing Jiang. 2009. Multi-task transfer learning for weakly-supervised relation extraction. 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*, pages 1012–1020.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

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)*, volume 1, pages 402–412.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

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.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1105–1116.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1858–1869.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:8.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2121–2130.

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 *Proceedings of the 26th International Conference on World Wide Web*, pages 1015–1024.

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*, pages 148–163.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *AAAI*.

Amit Singhal. 2012. Introducing the knowledge graph: things, not strings. *Official google blog*, 5.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706.

Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A hierarchical framework for relation extraction with reinforcement learning. In *AAAI*.

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

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):26–31.

Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. 2016. Supertagging with lstms. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237.

Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint extraction of entities and relations based on a novel graph scheme. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4461–4467.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *ICLR*.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 506–514.

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 *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1227–1236.