A Novel Hierarchical Binary Tagging Framework for Joint Extraction of Entities and Relations

Zhepei Wei[1], **Jianlin Su**[2], **Yue Wang**[3], **Yuan Tian**[1], **Yi Chang**[1]
[1]School of Artificial Intelligence, Jilin University
[2]Shenzhen Zhuiyi Technology Co., Ltd.
[3]School of Information and Library Science, University of North Carolina at Chapel Hill
weizp19@mails.jlu.edu.cn, bojonesu@wezhuiyi.com, wangyue@email.unc.edu, yuantian@jlu.edu.cn, yichang@jlu.edu.cn

Abstract

Extracting relational triples from unstructured text is crucial for large-scale knowledge graph construction. However, few existing works excel in solving the overlapping triple problem where multiple relational triples in the same sentence share the same entities. We propose a novel Hierarchical Binary Tagging (HBT) framework derived from a principled problem formulation. Instead of treating relations as discrete labels as in previous works, our new framework models relations as functions that map subjects to objects in a sentence, which naturally handles overlapping triples. Experiments show that the proposed framework already outperforms state-of-the-art methods even its encoder module uses a randomly initialized BERT encoder, showing the power of the new tagging framework. It enjoys further performance boost when employing a pretrained BERT encoder, outperforming the strongest baseline by 25.6 and 45.9 absolute gain in F1-score on two public datasets NYT and WebNLG, respectively. In-depth analysis on different types of overlapping triples shows that the method delivers consistent performance gain in all scenarios.

Introduction

The key ingredient of a knowledge graph is relational facts, most of which consist of two entities connected by a semantic relation. These facts are in the form of (subject, relation, object), or (s, r, o), referred to as relational triples. Extracting relational triples from natural language text is a crucial step towards constructing large-scale knowledge graphs.

Early works in relational triple extraction took a pipeline approach (Zelenko, Aone, and Richardella 2003; Zhou et al. 2005; Chan and Roth 2011). It first recognizes all entities in a sentence and then performs relation classification for each entity pair. Such an approach tends to suffer from the error propagation problem since errors in early stages cannot be corrected in later stages. To tackle this problem, subsequent works proposed joint extraction of entities and relations, among them are feature-based models (Yu and Lam 2010; Li and Ji 2014; Miwa and Sasaki 2014; Ren et al. 2017) and, more recently, neural network-based models (Zheng et al. 2017; Zeng et al. 2018; Fu, Li, and Ma 2019). By replacing manually constructed features with learned representations, neural network-based models have achieved considerable success in the triple extraction task.

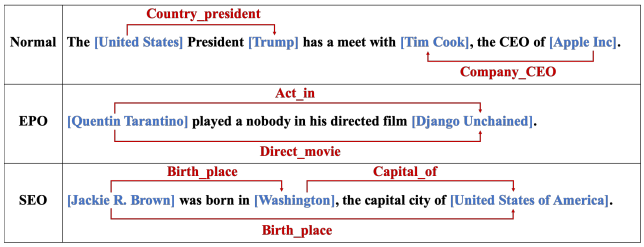


Figure 1: Examples of *Normal*, *EntityPairOverlap (EPO)* and *SingleEntityOverlap (SEO)* overlapping patterns.

However, most existing approaches cannot properly handle scenarios in which a sentence contains multiple relational triples that overlap with each other. Figure 1 illustrates these scenarios, where triples share one or two entities in a sentence. This *overlapping triple problem* directly challenges conventional sequence tagging schemes that assume each token bears only one tag (Zheng et al. 2017). It also brings significant difficulty to relation classification approaches where an entity pair is assumed to hold at most one relation (Katiyar and Cardie 2017). (Zeng et al. 2018) is the first to consider the overlapping triple problem. They proposed the categories for different overlapping patterns as shown in Figure 1 and proposed a sequence-to-sequence model with copy mechanism. (Fu, Li, and Ma 2019) also studied the overlapping triple problem and achieved further improvement with a model based on graph convolutional networks (GCNs).

Despite their success, previous works on extracting overlapping triples still leave much to be desired. Specifically, they all treat relations as discrete labels to be assigned to entity pairs. This formulation makes relation classification a difficult machine learning problem. For instance, when the same entity in the same context participates in multiple (in some cases more than five) valid relations, i.e. overlapping triples, the classifier needs considerable supervision to figure out what contextual information corresponds to one relation and not others. As a result, the extracted triples are usually incomplete and inaccurate.

In this work, we start with a principled formulation of re-

lational triple extraction right at the triple level. This gives rise to a general algorithmic framework that handles the overlapping triple problem by design. At the core of the framework is the fresh perspective that instead of treating relations as discrete labels on entity pairs, we can model relations as functions that map subjects to objects. More precisely, instead of learning relation classifiers $f(s, o) \rightarrow r$, we learn *relation-specific taggers* $f_r(s) \rightarrow o$, each of which recognizes the possible object(s) of a given subject under a specific relation; or returns no object, indicating there is no triple with the given subject and relation. Under this framework, triple extraction is a two-step process: first we identify all possible subjects in a sentence; then for each subject, we apply relation-specific taggers to simultaneously identify all possible relations and the corresponding objects.

We implement the above idea in an end-to-end hierarchical binary tagging (HBT) framework. It consists of a BERT-based encoder module, a subject tagging module, and a relation-specific object tagging module. Empirical experiments show that the proposed framework outperforms state-of-the-art methods by a large margin even when the BERT encoder is *not* pretrained, showing the superiority of the new framework itself. The framework enjoys a further large performance gain after adopting a pretrained BERT encoder, showing the importance of rich prior knowledge in triple extraction task.

The main contributions of this work are as follows:

1. We present a principled formulation of the relational triple extraction problem, which implies a general algorithmic framework for relational triple extraction that handles overlapping triple problem by design.
2. We instantiate the above framework as a novel hierarchical binary tagging model on top of a Transformer encoder. This allows the model to combine the power of the novel tagging framework with the prior knowledge in pretrained large-scale language models.
3. Extensive experiments on two public datasets show that the proposed framework overwhelmingly outperforms state-of-the-art methods, achieving 25.6 and 45.9 absolute gain in F1-score on the two datasets respectively. Detailed analyses show that our model gains consistent improvement in all scenarios.

Related Work

Extracting relational triples from unstructured natural language texts is a well-studied task in information extraction (IE). It is also an important step for the construction of large scale knowledge graph (KG) such as DBpedia (Auer et al. 2007), Freebase (Bollacker et al. 2008) and Knowledge Vault (Dong et al. 2014).

Early works (Mintz et al. 2009; Gormley, Yu, and Dredze 2015) address the task in a pipelined manner. They extract relational triples in two steps: 1) first run named entity recognition (NER) on the input sentence to identify all entities and then 2) run relation classification (RC) on pairs of extracted entities. The pipelined methods usually suffer from error propagation problems since the second step is unavoidably affected by the errors introduced in the first step. To ease these issues, many joint models that aim to learn entities and relations jointly have been proposed. Traditional joint models (Yu and Lam 2010; Li and Ji 2014; Miwa and Sasaki 2014; Ren et al. 2017) are feature-based, which heavily rely on feature engineering and require intensive manual efforts. To reduce manual work, recent studies have investigated neural network-based methods, which deliver state-of-the-art performance. However, most existing neural models (Miwa and Bansal 2016; Katiyar and Cardie 2017) achieve joint learning of entities and relations only through parameter sharing but not joint decoding. To obtain relational triples, they still have to pipeline the detected entity pairs to a relation classifier for identifying the relation of entities. The separated decoding setting leads to a separated training objective for entity and relation, which brings a drawback that the triple-level dependencies between predicted entities and relations cannot be fully exploited. Different from those works, (Zheng et al. 2017) achieves joint decoding by introducing a novel unified tagging scheme and convert the task of relational triple extraction to an end-to-end sequence tagging problem without need of NER or RC. Their proposed model can directly learn relational triples as a whole since the information of entities and relations is integrated into the unified tagging scheme.

Though joint models (with or without joint decoding) have been well studied, most previous works ignore the problem of overlapping relational triples. (Zeng et al. 2018) is the first to propose the overlapping problem and try to address it via a sequence-to-sequence (Seq2Seq) model with copy mechanism. Recently, (Fu, Li, and Ma 2019) also studies the problem and propose a graph convolutional networks (GCNs) based method. Despite their initial success, both methods suffer from cumbersome decoding processes.

Our proposed framework is based on a training objective that is carefully designed to directly model the relational triples as a whole, i.e., to learn both entities and relations through joint decoding. This makes it crucially different from previous works.

Hierarchical Binary Tagging Framework

The goal of relational triple extraction is to identify all possible (subject, relation, object) triples in a sentence, where some triples may share the same entities as subjects or objects. Towards this goal, we design the training objective at the triple level, and then decompose it down to entity and relation level. This is in contrast to previous approaches (Zeng et al. 2018; Fu, Li, and Ma 2019) where the training objective is defined separately for entities and relations without explicitly modeling their integration at the triple level.

Formally, given annotated sentence x_j from training set D and a set of potentially overlapping triples $T_j = \{(s, r, o)\}$ in x_j, the proposed framework aims to maximize the data likelihood of the training set D:

$$\prod_{j=1}^{|D|}\left[\prod_{(s,r,o)\in T_j} p((s,r,o)|x_j)\right] \quad (1)$$

$$=\prod_{j=1}^{|D|}\left[\prod_{s\in T_j} p(s|x_j)\prod_{(r,o)\in T_j|s} p((r,o)|s,x_j)\right] \quad (2)$$

$$=\prod_{j=1}^{|D|}\left[\prod_{s\in T_j} p(s|x_j)\prod_{r\in T_j|s} p_r(o|s,x_j)\prod_{r\in R\setminus T_j|s} p_r(o_\varnothing|s,x_j)\right]. \quad (3)$$

Here we slightly abuse the notation T_j. $s \in T_j$ denotes a subject appearing in the triples in T_j. $T_j|s$ is the set of triples led by subject s in T_j. $(r,o) \in T_j|s$ is a (r,o) pair in the triples led by subject s in T_j. R is the set of all possible relations. $R\setminus T_j|s$ denotes all relations except those led by s in T_j. $o_\varnothing$ denotes a "null" object (explained below).

Eq. (2) applies the chain rule of probability. Eq. (3) exploits the crucial fact that for a given subject s, any relation relevant to s (those in $T_j|s$) would lead to corresponding objects in the sentence, and all other relations would necessarily have no object in the sentence, i.e. a "null" object.

This formulation provides several benefits. First, since the data likelihood starts at the triple level, optimizing this likelihood corresponds to directly optimizing the final evaluation criteria at the triple level. Second, by making no assumption on how multiple triples may share entities in a sentence, it handles overlapping triple problem *by design*. Third, the decomposition in Eq. (3) inspires a novel tagging scheme for triple extraction: we learn a subject tagger $p(s|x_j)$ that recognizes subject entities in a sentence; and for each relation r, we learn an object tagger $p_r(o|s,x_j)$ that recognize relation-specific objects for a given subject. In this way we can model each relation as a function that maps subjects to objects, as opposed to classifying relations for (*subject*,*object*) pairs.

Indeed, this novel tagging scheme allows us to extract multiple triples at once: we simply run the subject tagger to find all possible subjects in the sentence, and then for each subject found, apply relation-specific object taggers to find all relevant relations and the corresponding objects.

The key components in the above general framework, i.e., the subject tagger and relation-specific object taggers, can be instantiated in many ways. In this paper, we instantiate them as binary taggers on top of a deep bidirectional Transformer BERT (Devlin et al. 2019). We describe its details below.

BERT Encoder

The encoder module extracts feature information $\mathbf{x}_j$ from sentence x_j, which will feed into subsequent tagging modules[1]. We employ a pretrained BERT model (Devlin et al. 2019) to encode the context information.

Here we briefly review BERT, a multi-layer bidirectional Transformer based language representation model. It is designed to learn deep representations by jointly conditioning on both left and right context of each word and it has recently been proven surprisingly effective in many downstream tasks. Specifically, it is composed of a stack of N identical Transformer blocks. We denote the Transformer block as $Trans(\mathbf{x})$, in which $\mathbf{x}$ represents the input vector. The detailed operations are as follows:

$$\mathbf{h}_0 = \mathbf{S}\mathbf{W}_s + \mathbf{W}_p \quad (4)$$

$$\mathbf{h}_\alpha = Trans(\mathbf{h}_{\alpha-1}), \alpha \in [1,N] \quad (5)$$

Where $\mathbf{S}$ is the matrix of one-hot vectors of sub-words[2] indices in the input sentence, $\mathbf{W}_s$ is the sub-words embedding matrix, $\mathbf{W}_p$ is the positional embedding matrix where p represents the position index in the input sequence, $\mathbf{h}_\alpha$ is the hidden state vector, i.e., the context representation of input sentence at α-th layer and N is the number of Transformer blocks. Note that in our work the input is a single text sentence instead of sentence pair, hence the segmentation embedding as described in original BERT paper was not taken into account in Eq. (4). For a more comprehensive description of the Transformer structure, we refer readers to (Vaswani et al. 2017) and the excellent guide "The Annotated Transformer"[3].

Hierarchical Decoder

Now we describe our instantiation of the new hierarchical tagging scheme inspired by the previous formulation. The basic idea is to extract triples in two steps. First, we detect subjects from the input sentence. Then for each candidate subject, we check all possible relations to see if a relation can associate objects in the sentence with that subject. Corresponding to the two steps, the hierarchical decoder consists of two modules as illustrated in Figure 2: a subject tagger; and a set of relation-specific object taggers.

Subject Tagger The low level tagging module is designed to recognize all possible subjects in the input sentence by directly decoding the encoded vector $\mathbf{h}_N$ produced by the N-layer BERT encoder. More precisely, it adopts two identical binary classifiers to detect the start and end position of subjects respectively by assigning each token a binary tag (0/1) that indicates whether the current token corresponds to a start or end position of a subject. The detailed operations of the subject tagger on each token are as follows:

$$p_i^{start\_s} = \sigma(\mathbf{W}_{start}\mathbf{x}_i + \mathbf{b}_{start}) \quad (6)$$

$$p_i^{end\_s} = \sigma(\mathbf{W}_{end}\mathbf{x}_i + \mathbf{b}_{end}) \quad (7)$$

where $p_i^{start\_s}$ and $p_i^{end\_s}$ represent the probability of identifying the i-th token in the input sequence as the start and end position of a subject respectively. The corresponding token will be assigned with a tag 1 if the probability exceeds a certain threshold or with a tag 0 otherwise. $\mathbf{x}_i$ is the encoded representation of the i-th token in the input sequence, i.e., $\mathbf{x}_i = \mathbf{h}_N[i]$, $\mathbf{W}_{(\cdot)}$ represents the trainable weight, $\mathbf{b}_{(\cdot)}$ is the bias and σ is the sigmoid activation function.

[1] This paper uses boldface letters to denote vectors and matrices.

[2] We use Wordpiece embeddings (Wu et al. 2016) to represent words in vector space, which means each word in the input sentence will be split to fine-grained tokens, i.e., sub-words.

[3] http://nlp.seas.harvard.edu/2018/04/03/attention.html

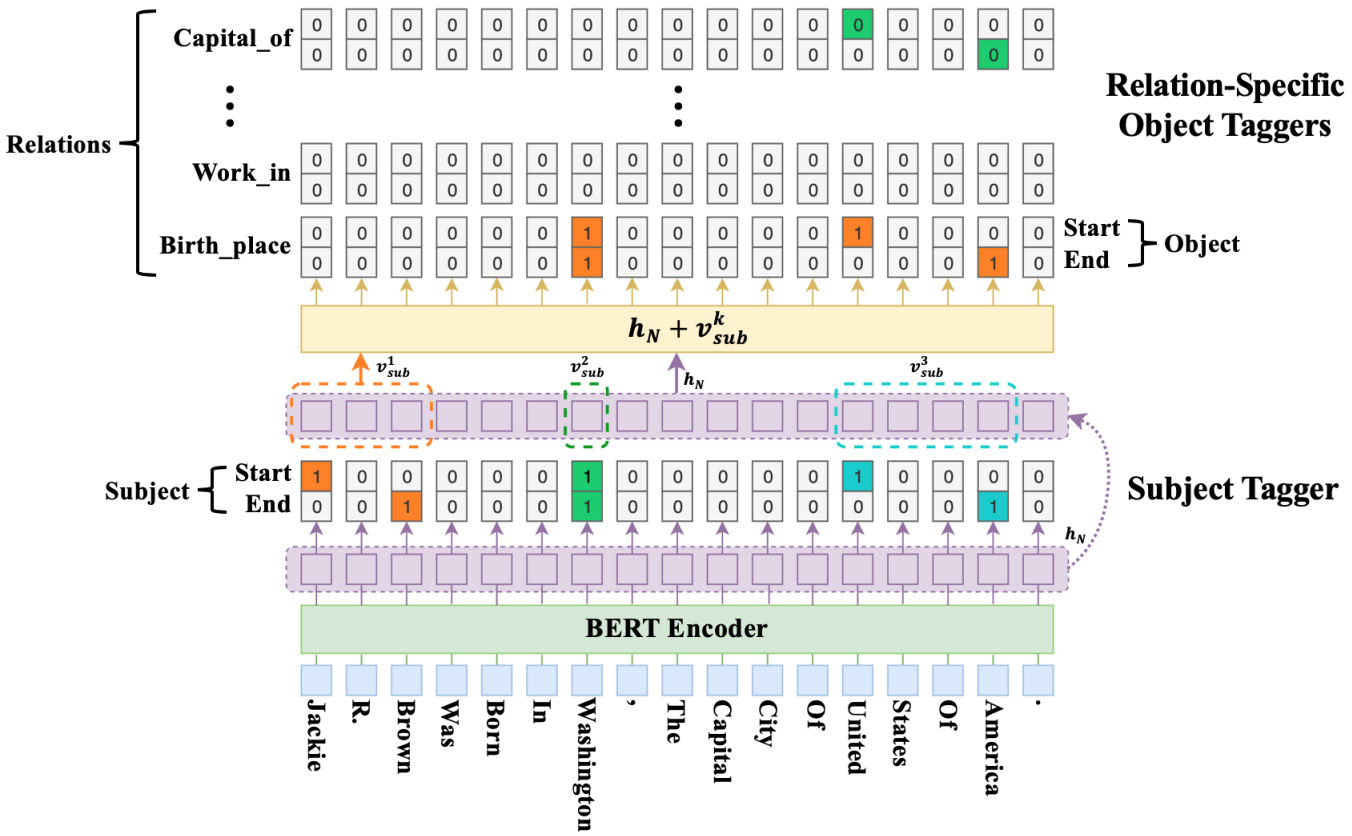


Figure 2: An overview of the proposed hierarchical binary tagging (HBT) framework structure. In this example, there are three candidate subjects detected at the low level, while the presented 0/1 tags at high level are specific to the first subject *Jackie R. Brown*, i.e., a snapshot of the iteration state when $k = 1$ is shown as above. For the subsequent iterations ($k = 2, 3$), the results at high level will change, reflecting different triples detected. For instance, when $k = 2$, the high-level orange (green) blocks will change to 0 (1), respectively, reflecting the relational triple *(Washington, Capital_of, United States Of America)* led by the second candidate subject *Washington*.

The subject tagger optimizes the following likelihood function to identify the span of subject s given a sentence representation $\mathbf{x}$:

$$p_\theta(s|\mathbf{x}) = \prod_{t\in\{start\_s, end\_s\}} \prod_{i=1}^{L} \left(p_i^t\right)^{\mathbf{1}\{y_i^t=1\}} \left(1-p_i^t\right)^{\mathbf{1}\{y_i^t=0\}} . \tag{8}$$

The parameters $\theta = \{\mathbf{W}_{start}, \mathbf{b}_{start}, \mathbf{W}_{end}, \mathbf{b}_{end}\}$. L is the length of the sentence. $\mathbf{1}\{z\} = 1$ if z is true and 0 otherwise. $y_i^{start\_s}$ is the binary tag of subject start position for the i-th token in $\mathbf{x}$, and $y_i^{end\_s}$ indicates the subject end position.

For multiple subjects detection, we adopt the nearest start-end pair match principle to decide the span of any subject based on the results of the start and end position taggers. For example, as shown in Figure 2, the nearest end token to the first start token "Jackie" is "Brown", hence the detected result of the first subject span will be "Jackie R. Brown". Noticeably, to match an end token for a given start token, we only consider tokens whose position is behind the position of the given token. Such match strategy is able to maintain the integrity of any entity span if the start and end positions are both correctly detected due to the natural continuity of any entity span in a given sentence.

Relation-specific Object Taggers High level tagging module simultaneously identifies the objects as well the involved relations with respect to the subjects obtained at lower level. As Figure 2 shows, it consists of a set of relation-specific object taggers with the same structure as subject tagger in low level module for all possible relations. All object taggers will identify the corresponding object(s) for each detected subject at the same time. Different from subject tagger directly decoding the encoded vector $\mathbf{h}_N$, the relation-specific object tagger takes the subject features into account as well. The detailed operations of the relation-specific object tagger on each token are as follows:

$$p_i^{start\_o} = \sigma(\mathbf{W}^r_{start}(\mathbf{x}_i + \mathbf{v}^k_{sub}) + \mathbf{b}^r_{start}) \tag{9}$$

$$p_i^{end\_o} = \sigma(\mathbf{W}^r_{end}(\mathbf{x}_i + \mathbf{v}^k_{sub}) + \mathbf{b}^r_{end}) \tag{10}$$

where $p_i^{start\_o}$ and $p_i^{end\_o}$ represent the probability of identifying the i-th token in the input sequence as the start and

end position of a object respectively, and $\mathbf{v}_{sub}^k$ represents the encoded representation vector of the k-th subject detected in low level module.

For each subject, we iteratively apply the same decoding process on it. Note that the subject is usually composed of multiple tokens, to make the additions in Eq. (9) and Eq. (10) possible, we need to keep the dimension of two vectors consistent. To do so, we take the averaged vector representation of all the tokens that the k-th subject contains as $\mathbf{v}_{sub}^k$.

The object tagger for relation r optimizes the following likelihood function to identify the span of object o given a sentence representation $\mathbf{x}$ and a subject s:

$$p_{\phi_r}(o|s,\mathbf{x}) = \prod_{t\in\{start\_o,end\_o\}} \prod_{i=1}^{L} \left(p_i^t\right)^{\mathbf{1}\{y_i^t=1\}} \left(1-p_i^t\right)^{\mathbf{1}\{y_i^t=0\}} . \quad (11)$$

The parameters $\phi_r = \{\mathbf{W}_{start}^r, \mathbf{b}_{start}^r, \mathbf{W}_{end}^r, \mathbf{b}_{end}^r\}$. $y_i^{start\_o}$ is the binary tag of object start position for the i-th token in $\mathbf{x}$, and $y_i^{end\_o}$ is the tag of object end position for the i-th token. For a "null" object $o_\varnothing$, the tags $y_i^{start\_o_\varnothing} = y_i^{end\_o_\varnothing} = 0$ for all i.

Note that in the high level tagging module, the relation is also decided by the output of object taggers. For example, the relation "Work_in" does not hold between the detected subject "Jackie R. Brown" and candidate object "Washington". Therefore, the object tagger for relation "Work_in" will not identify the span of "Washington", i.e., the output of both start and end position are all zeros as shown in Figure 2. In contrast, the relation "Birth_place" exits between "Jackie R. Brown" and "Washington", so the corresponding object tagger outputs the span of the candidate object "Washington". In this setting, the high level module is capable of simultaneously identifying the relations and objects with regard to the subjects detected in low level module.

Data Log-likelihood Objective Taking log of Eq. (3):

$$J(\Theta) = \sum_{j=1}^{|D|} \left[\sum_{s\in T_j} \log p_\theta(s|\mathbf{x}_j) + \sum_{r\in T_j|s} \log p_{\phi_r}(o|s,\mathbf{x}_j) + \sum_{r\in R\setminus T_j|s} \log p_{\phi_r}(o_\varnothing|s,\mathbf{x}_j) \right] . \quad (12)$$

where parameters $\Theta = \{\theta, \{\phi_r\}_{r\in R}\}$. $p_\theta(s|\mathbf{x})$ is defined in (8) and $p_{\phi_r}(o|s,\mathbf{x})$ is defined in (11). We train the model by maximizing $J(\Theta)$ through Adam stochastic gradient descent over shuffled mini-batches (Kingma and Ba 2014).

Experiments

Experimental Setting

Datasets and Evaluation Metrics We evaluate the proposed HBT framework on two public datasets NYT (Riedel, Yao, and McCallum 2010) and WebNLG (Gardent et al. 2017). NYT dataset was originally produced by distant supervision method. It consists of 1.18M sentences with 24 predefined relation types. WebNLG dataset was originally created for Natural Language Generation (NLG) tasks and adapted by (Zeng et al. 2018) for relational triple extraction task. It contains 246 predefined relation types. The sentences in both datasets usually contain multiple relational triples, thus NYT and WebNLG datasets are widely used to serve as the preferred testbed for evaluating model effectiveness in extracting overlapping relational triples. For fair comparison with previous works, we directly conduct experiments on the preprocessed datasets released by (Zeng et al. 2018), in which NYT contains 56195 sentences for training, 5000 sentences for validation, and 5000 sentences for test, and WebNLG contains 5019 sentences for training, 500 sentences for validation and 703 sentences for test. According to different overlapping patterns of relational triples, we split the sentences into three categories, namely, *Normal*, *EntityPairOverlap (EPO)* and *SingleEntityOverlap (SEO)* for detailed experiments on different types of overlapping relational triples. The statistics of the two datasets are described in Table 1.

Category	NYT		WebNLG	
	Train	Test	Train	Test
Normal	37013	3266	1596	246
EPO	9782	978	227	26
SEO	14735	1297	3406	457
ALL	56195	5000	5019	703

Table 1: Statistics of datasets. Note that a sentence can belong to both *EPO* class and *SEO* class.

Following previous works, we use the standard Precision (Prec.), Recall (Rec.) and F1-score as the evaluation metrics. An extracted relational triple *(subject, relation, object)* is regarded as correct only if the relation and the heads of both subject and object are all correct.

Implementation Details We adopt mini-batch mechanism to train our model with batch size as 8; the learning rate is set to 0.001; the hyper-parameters are determined on the validation set. We also adopt early stopping mechanism to prevent the model from over-fitting. Specifically, we stop the training process when the performance on validation set does not gain any improvement for at least 7 consecutive epochs. The number of stacked bidirectional Transformer blocks N is 12 and the size of hidden state $\mathbf{h}_N$ is 768. The pre-trained BERT model we used is [BERT-Base, Cased][4],which contains 110M parameters.

For fair comparison, the max length of input sentence to our model is set to 100 words as previous works (Zeng et al. 2018; Fu, Li, and Ma 2019) suggest. We did not tune the threshold for both start and end position taggers to predict tag 1, but heuristically set the threshold to 0.5 as default. The performance might be better after carefully tuning the threshold, however it is beyond the research scope of this paper.

[4]Available at: https://storage.googleapis.com/bert_models/2018_10_18/cased_L-12_H-768_A-12.zip

Method	NYT			WebNLG		
	Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
NovelTagging (Zheng et al. 2017)	62.4	31.7	42.0	52.5	19.3	28.3
CopyR$_{OneDecoder}$ (Zeng et al. 2018)	59.4	53.1	56.0	32.2	28.9	30.5
CopyR$_{MultiDecoder}$ (Zeng et al. 2018)	61.0	56.6	58.7	37.7	36.4	37.1
GraphRel$_{1p}$ (Fu, Li, and Ma 2019)	62.9	57.3	60.0	42.3	39.2	40.7
GraphRel$_{2p}$ (Fu, Li, and Ma 2019)	63.9	60.0	61.9	44.7	41.1	42.9
HBT$_{random}$	84.7	72.3	78.0	67.9	40.4	50.6
HBT	**89.7**	**85.4**	**87.5**	**89.5**	**88.0**	**88.8**

Table 2: Results of different methods on NYT and WebNLG datasets.

Experimental Result

Compared Methods We compare our model with three strong state-of-the-art models, namely, **NovelTagging** (Zheng et al. 2017), **CopyR** (Zeng et al. 2018) and **GraphRel** (Fu, Li, and Ma 2019). The reported results for the above baselines are directly copied from the original published literature.

To evaluate the impact of introducing the pre-trained BERT model as the encoder in our HBT framework, we conduct ablation tests. **HBT**$_{random}$ is the HBT framework where the parameters of BERT encoder are randomly initialized and the weights are learned from scratch; **HBT** is the full-fledged framework using pre-trained BERT weights.

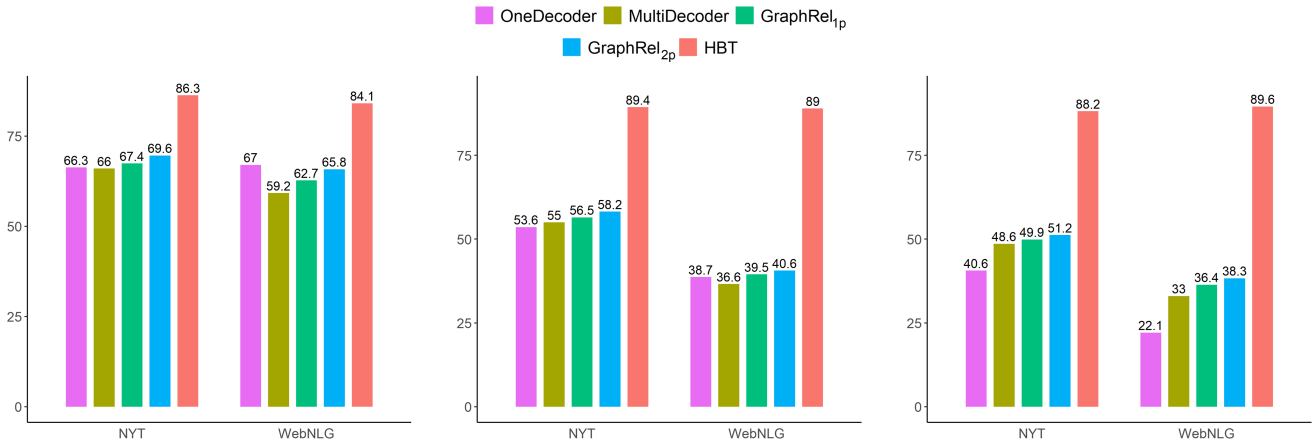
Main Results Table 2 shows the results of different baselines for joint extraction of entities and relations on two datasets. The HBT model overwhelmingly outperforms all the baselines in terms of all three evaluation metrics and achieves encouraging 25.6% and 45.9% improvements in F1-score over the best state-of-the-art method (Fu, Li, and Ma 2019) on NYT and WebNLG datasets respectively. Even without taking advantage of the pre-trained language model, the HBT$_{random}$ method is still competitive to existing state-of-the-art models. This validates the effectiveness of the proposed hierarchical decoder that adopts a novel binary tagging scheme. The performance improvements from HBT$_{random}$ to HBT highlight the importance of introducing prior knowledge in pre-trained BERT model into the framework.

We can also observe from the table that there is a significant gap between the performances on NYT and WebNLG datasets for existing models and we believe this gap is due to their drawbacks in dealing with overlapping triples. More precisely, as presented in Table 1, we can see that NYT dataset is mainly comprised of *Normal* class sentences while the majority of sentences in WebNLG dataset belong to *EPO* and *SEO* classes. Such inconsistent data distribution of two datasets leads to a comparatively better performance on NYT and a worse performance on WebNLG for all the baselines, exposing their drawbacks in extracting overlapping relational triples. In contrast, the HBT model achieves a stable and competitive performance on both NYT and WebNLG datasets, demonstrating the effectiveness of the proposed framework in solving the overlapping problem.

Detailed Results on Different Types of Sentences To further study the capability of the proposed HBT framework in extracting overlapping relational triples, we conduct two extended experiments on different types of sentences and compare the performance with previous works.

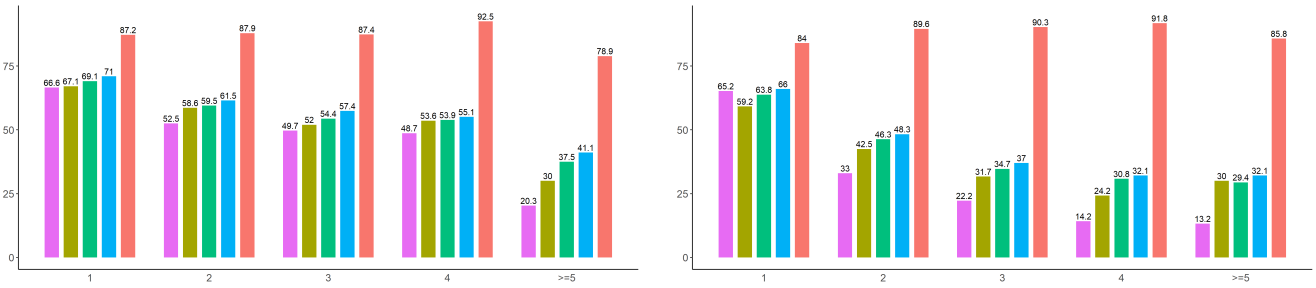
The detailed results on three different overlapping patterns are presented in Figure 3. It can be seen that the performance of baseline models on *Normal*, *EPO* and *SEO* present a decreasing trend, reflecting the increasing difficulty of extracting relational triples from sentences with different overlapping patterns. That is, among the three overlapping patterns, *Normal* class is the easiest pattern to extract while *EPO* and *SPO* classes are the relatively harder ones for baseline models to extract. In contrast, the proposed HBT framework attains consistently excellent performance over all three overlapping patterns, especially for those hard patterns. We also compare the model's capability of extracting relational triples from sentences that contains different number of triples. We split the sentences into five classes and Figure 4 shows the results. Noticeably, our HBT model achieves excellent performance over all five classes. Though it's not surprising to find that the performances of all the baselines decrease with the increasing number of relational triples that a sentence contains, some patterns still can be observed from the performance changes of different models. On one hand, compared to previous works that devote to solving the overlapping problem in relational triple extraction, our model suffers the least from the increasing complexity of the input sentence. In fact, the performance of the proposed HBT framework increases somewhat when the complexity of the input sentence keeps growing under a certain upper limit, indicating that our model is more suitable for complicated scenarios than the baselines.

Both of these experiments validate the superiority of the proposed hierarchical binary tagging framework in extracting multiple (possibly overlapping) relational triples from complicated sentences compared to existing methods. Previous works have to explicitly predict all possible relation types contained in a given sentence, which is quite a challenging task, and thus many relations are missing in their extracted results. In contrast, our HBT model side-steps the prediction of relation types and tends to extract as many relational triples as possible from a given sentence. We attribute this to the relation-specific object tagger setting in high-level



(a) F1 of Normal Class (b) F1 of EntityPairOverlap Class (c) F1 of SingleEntityOverlap Class

Figure 3: F1-score of extracting relational triples from sentences with different overlapping pattern.



(a) F1 of Different #Triples (NYT) (b) F1 of Different #Triples (WebNLG)

Figure 4: F1-score of extracting relational triples from sentences containing different number of triples (denoted by #Triples).

Element	NYT			WebNLG		
	Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
(E1, R)	93.7	88.3	90.9	92.5	88.7	90.5
(E2, R)	93.0	87.4	90.1	92.4	89.2	90.8
(E1, E2)	89.3	85.9	87.6	91.9	90.3	91.1
(E1, R, E2)	89.7	85.4	87.5	89.5	88.0	88.8

Table 3: Results on relational triple elements.

tagging module of the hierarchical decoder that considers all the relation types simultaneously.

Error Analysis In order to explore the factors that affect the extracted relational triples of our HBT model, we analyze the performance on predicting different elements of the triple, i.e., *(E1, R)*, *(E2, R)* and *(E1, E2)* where *E1* represents the subject entity, *E2* represents the object entity and *R* represents the relation between them. Table 3 shows the results on different relational triple elements. For both NYT and WebNLG, the performance on extracting *(E1, R)* is comparable to that on extracting *(E2, R)*, demonstrating the effectiveness of our proposed framework in identifying the span of both subject and object entity mention. However, the performance on extracting *(E1, E2)* shows an opposite trend between the two datasets, which we attribute to the inconsistent data distributions of the two datasets as detailed above.

Conclusion

In this paper, we propose a novel hierarchical binary tagging (HBT) framework for joint extraction of entities and relations. Benefiting from the novel tagging scheme, our model can efficiently extract multiple relational triples from sentences, without suffering from the overlapping problem. We conduct extensive experiments on two widely used datasets to validate the effectiveness of the proposed HBT framework. Experimental results show that our model overwhelmingly outperforms state-of-the-art baselines over different scenarios, especially on the extraction of overlapping relational triples. Moreover, our model is also a general framework for information extraction (IE). In the future, we will further validate the effectiveness of the proposed hierarchical binary tagging framework in other similar IE tasks, such

as joint event detection and arguments extraction.

References

[Auer et al. 2007] Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; and Ives, Z. 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*, 722–735.

[Bollacker et al. 2008] Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 1247–1250.

[Chan and Roth 2011] Chan, Y. S., and Roth, D. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, 551–560. Association for Computational Linguistics.

[Devlin et al. 2019] Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

[Dong et al. 2014] Dong, X.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; and Zhang, W. 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*, 601–610.

[Fu, Li, and Ma 2019] Fu, T.-J.; Li, P.-H.; and Ma, W.-Y. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[Gardent et al. 2017] Gardent, C.; Shimorina, A.; Narayan, S.; and Perez-Beltrachini, L. 2017. Creating training corpora for nlg micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 179–188.

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

[Katiyar and Cardie 2017] Katiyar, A., and Cardie, C. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 917–928.

[Kingma and Ba 2014] Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Li and Ji 2014] Li, Q., and Ji, H. 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, 402–412.

[Mintz et al. 2009] Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 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*, 1003–1011.

[Miwa and Bansal 2016] Miwa, M., and Bansal, M. 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, 1105–1116.

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

[Ren et al. 2017] Ren, X.; Wu, Z.; He, W.; Qu, M.; Voss, C. R.; Ji, H.; Abdelzaher, T. F.; and Han, J. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of the 26th International Conference on World Wide Web*, 1015–1024.

[Riedel, Yao, and McCallum 2010] Riedel, S.; Yao, L.; and McCallum, A. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 148–163.

[Vaswani et al. 2017] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

[Wu et al. 2016] Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

[Yu and Lam 2010] Yu, X., and Lam, W. 2010. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, 1399–1407. Association for Computational Linguistics.

[Zelenko, Aone, and Richardella 2003] Zelenko, D.; Aone, C.; and Richardella, A. 2003. Kernel methods for relation extraction. *Journal of machine learning research* 3(Feb):1083–1106.

[Zeng et al. 2018] Zeng, X.; Zeng, D.; He, S.; Liu, K.; and Zhao, J. 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, 506–514.

[Zheng et al. 2017] Zheng, S.; Wang, F.; Bao, H.; Hao, Y.; Zhou, P.; and Xu, B. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceed-*

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

[Zhou et al. 2005] Zhou, G.; Su, J.; Zhang, J.; and Zhang, M. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL05)*, 427–434.