MrMep: Joint Extraction of Multiple Relations and Multiple Entity Pairs Based on Triplet Attention

Jiayu Chen, Caixia Yuan, Xiaojie Wang and **Ziwei Bai**
Center of Intelligence Science and Technology
School of Computer Science
Beijing University of Posts and Telecommunications
{jiayuchen, yuancx, xjwang, bestbzw}@bupt.edu.cn

Abstract

This paper focuses on how to extract multiple relational facts from unstructured text. Neural encoder-decoder models have provided a viable new approach for jointly extracting relations and entity pairs. However, these models either fail to deal with entity overlapping among relational facts, or neglect to produce the whole entity pairs. In this work, we propose a novel architecture that augments the encoder and decoder in two elegant ways. First, we apply a binary CNN classifier for each relation, which identifies all possible relations maintained in the text, while retaining the target relation representation to aid entity pair recognition. Second, we perform a multi-head attention over the text and a triplet attention with the target relation interacting with every token of the text to precisely produce all possible entity pairs in a sequential manner. Experiments on three benchmark datasets show that our proposed method successfully addresses the multiple relations and multiple entity pairs even with complex overlapping and significantly outperforms the state-of-the-art methods. All source code and documentations are available at https://github.com/chenjiayu1502/MrMep.

1 Introduction

Extracting relational facts from unstructured text is a significant step in building large-scale knowledge graphs. A relational fact is typically represented as a triplet $<h, r, t>$ where h represents a head entity, t represents a tail entity, and r is a relation that connects h to t.

A number of neural models for extracting relational facts have been developed in recent years, and the most successful models all have one thing in common: they extract both relation and its corresponding entity pairs in a manner of joint learning (Zheng et al., 2017; Zeng et al., 2018; Takanobu et al., 2019). However, jointly extracting relation and entity pairs is far from a trivial task due that there might exist more than one relation within the text, while each target relation might correspond to more than one entity pair. A more challenging aspect of this problem is that there might exist complex overlapping among different triplets. As exemplified in Figure 1, the entity "$Indonesia$" overlaps in all triplets within the text, while $<Indonesia, leaderName, Jusuf\ Kalla>$ and $<Indonesia, leaderName, Joko\ Widodo>$ have the same relation type "$leaderName$", $<Bakso, region, Indonesia>$ and $<Bakso, country, Indonesia>$ share the same head and tail entities.

A well-defined relational fact extraction task aims to detect all possible relation types in the text, and extract all candidate entity pairs for each target relation type, while taking into account the complicated overlapping among the triplets.

Zheng et al. (2017) propose a tagging mechanism (referred as Tagging thereafter) to transform the relational fact extraction into a sequence labeling task by injecting the information of relation type and entity position into tags. In this paradigm, Tagging model assigns a unique label to each word, which fails to extract triplets with overlapped entities or even overlapped entity pairs.

Zeng et al. (2018) model the triplet extraction using sequence-to-sequence learning with copy mechanism (referred as CopyR thereafter) that is often used in sentence generation (Gu et al., 2016; He et al., 2017). Takanobu et al. (2019) apply a hierarchical framework with reinforcement learning, which decomposes triplet extraction into a high-level task for relation detection and a low-level task for entity extraction (referred as HRL thereafter). Both CopyR and HRL determine a relation type each time and detect an entity pair for it, then

Proceedings of the 23rd Conference on Computational Natural Language Learning, pages 593–602
Hong Kong, China, November 3-4, 2019.

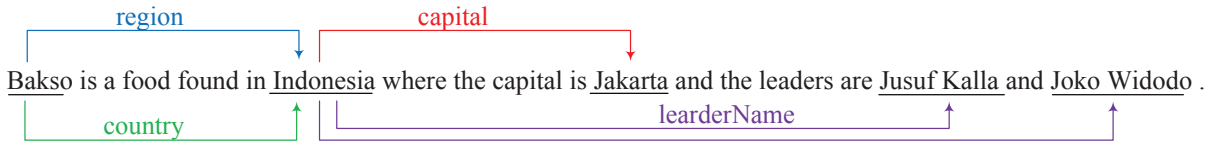


Figure 1: An example of multiple triplets. Entities are underlined, the relation types between entity pairs are connected by colored lines marked with relation types.

repeat the process to extract all triplets. However, to extract multiple entity pairs for a relation type, both CopyR and HRL have to repeatedly predict the relation type in multiple passes, which is computationally inefficient.

In this work, we propose, with simplicity and effectiveness in mind, a novel approach for jointly extracting Multiple Relations and Multiple Entity Pairs (MrMep). MrMep utilizes a triplet attention to exploit connections between relation and its corresponding entity pairs. It first predicts all possible relations, then for each target relation, it uses a variant of the pointer network (Vinyals et al., 2015) to generate boundaries (START/END positions) of all head and tail entities in a sequential manner, whereby the model generates all possible entity pairs as answers. In this way, for each candidate relation type, relation detection only needs to be performed one time, and then all possible entity pairs can be extracted for it, avoiding the repeated process of relation identification that is adopted in both CopyR and HRL. Moreover, we approach entity overlapping problems by a variant of the pointer network. It can sequentially generate entity boundaries in arbitrary position within the text. Therefore, it allows entities freely to participate in different triplets.

Our contributions can be summarized as follows:

- We propose MrMep, a novel neural method which firstly extracts all possible relations and then extracts all possible entity pairs for each target relation, while the two procedures are packed together into a joint model and are trained in a joint manner.
- MrMep uses a triplet attention to strengthen the connections among relation and entity pairs, and is computationally efficient for sophisticated overlapped triplets even with lightweight network architecture.
- Through extensive experiments on three benchmark datasets, we demonstrate MrMep's effectiveness over the most competitive state-of-the-art approach by 7.8%, 7.0% and 9.9% improvement respectively in F1 scores.

2 Related Work

Traditional pipelined approaches (Chan and Roth, 2011; Zelenko et al., 2003; Lin et al., 2016) separately design two subtasks of relation identification and entity pair extraction, ignoring the connection between them (Li and Ji, 2014) and suffering from error propagation.

To address the above problems, several joint models have been proposed. Early work (Li and Ji, 2014; Yu and Lam, 2010; Miwa and Sasaki, 2014) builds the connections between relation identification and entity recognition by designing multiple ingenious features, which needs complicated feature engineering. Recently, with the success of deep learning on many NLP tasks, several pieces of work (Miwa and Bansal, 2016; Gupta et al., 2016; Zhang et al., 2017) jointly model the interaction between the two subtasks based on neural networks, which they firstly apply RNN or CNN to encode the text, then treat entity recognition as a sequence labeling task and regard relation extraction as a multi-class classification problem. Zheng et al. (2017) formulate joint triplet extraction problem to the task of sequence labeling, by deliberately designing a tagging schema that injects information of relation type and entity position into tags. However, due to the limitation that a word can only be assigned a unique label in sequence labeling task and unique relation type can be assigned to each entity pair in multi-class classification problem, the above methods cannot fully address the triplet overlapping problem.

Most recent work aims at further exploring the overlapping triplet extraction. Zeng et al. (2018) propose CopyR, a joint model based on copy mechanism to convert the joint extraction task into a triplet generation task. For simplification, CopyR only copies the last word of the

entity, thereafter it cannot extract the whole entities consisting of multiple words. Takanobu et al. (2019) propose a hierarchical reinforcement learning framework that decomposes the joint extraction task into a high-level task for relation detection and a low-level task for entity extraction. However, both CopyR and HRL generate each relation type along with one entity pair at each time. In this paradigm, both of them have to perform the same relation detection multiple times to explore all possible entity pairs, which is computationally inefficient and principled not graceful enough.

We propose MrMep, a novel joint extraction framework that not only can jointly model triplet extraction task, but also efficiently extract various overlapped triplets. First, MrMep executes multiple relation classifiers by adopting a binary classifier for each relation type. Second, MrMep performs variable-length entity pair predictor. It utilizes a pointer network (Vinyals et al., 2015) alike way to generate the START/END positions for all the candidate head and tail entities in a sequential fashion, which is tracked by an LSTM decoder until it produces a word position beyond the text boundary. Inspired by machine reading and comprehending models (Wang and Jiang, 2016; Seo et al., 2017), we use the relation as a query, and conduct interaction between query and text via a triplet attention with relation peeking all possible entity pairs. We discuss two different principled ways to perform the above triplet attention, and demonstrate the versatility and effectiveness of our methods on both English and Chinese relational fact extraction benchmarks.

The main difference between our models and CopyR and HRL is that ours has the advantage of learning a strategic decoder using triplet attention that can model the interaction between relation and entity pairs. By letting the decoder only predict entity pairs, we relieve it from the burden of having to generate relation types repeatedly.

Das et al. (2019)'s work is similar in spirit to our work in that we both use a machine reading comprehension framework to extract relations and entities. But our approach differs from that of Das et al. (2019): their work aims at extracting newest state from procedural text which mainly describes the entity state at different steps, while our work is proposed for extracting multiple triplets from arbitrary unstructured text. Another similar approach is introduced by Roth et al. (2019), which performs relation argument extraction given a query entity and relation, essentially not for joint triplet extraction.

3 The Approach

3.1 Overview

For relational fact extraction task, the input is a text paragraph with n words $S = [w_1, ..., w_n]$. Let $\mathcal{R}$ be the set of predefined relation types. The task is to predict all possible triplets $<e_i, r_{ij}, e_j>$ maintained in S, where e_i, e_j are sequences of tokens denoting head entity and tail entity respectively, and $r_{ij} \in \mathcal{R}$ is the relation type that connects e_i to e_j.

Figure 2 shows an overview architecture of the proposed MrMep. It consists of three main parts: Encoder, Multiple Relation Classifiers and Variable-length Entity Pair Predictor. The encoder preprocesses the source text and extracts the sequence-level features using a Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), the multiple relation classifiers predict all possible relations maintained in S, and the variable-length entity pair predictor sequentially generates all possible entity pairs for each possible relation type.

3.2 Encoder

Given a text paragraph $S = [w_1, ..., w_n]$, we first embed it to obtain text embedding $E \in \mathbb{R}^{n \times d_e}$, where n is the length of words in text, d_e denotes the dimension of word embedding pretrained using Glove (Pennington et al., 2014). Then an LSTM is used to learn the token representation X_i for each word w_i:

$$X_i = LSTM_{encoder}(E_i, X_{i-1}), \quad (1)$$

where X_{i-1}, $X_i \in \mathbb{R}^d$ denotes word vectors for word w_{i-1} and w_i, yielding the text representation matrix $X = [X_1, ...X_n] \in \mathbb{R}^{n \times d}$.

The LSTM maps the variable length input sequences to a fixed-sized vector, and uses the last hidden state $X_n \in \mathbb{R}^d$ as the representation vector of the text.

3.3 Multiple Relation Classifiers

The relation classifier aims to identify relation types contained in the text. Since a text may contain multiple relations, inspired by the idea of multi-label classification, we design M CNN (Kim, 2014) based binary classifiers, respectively

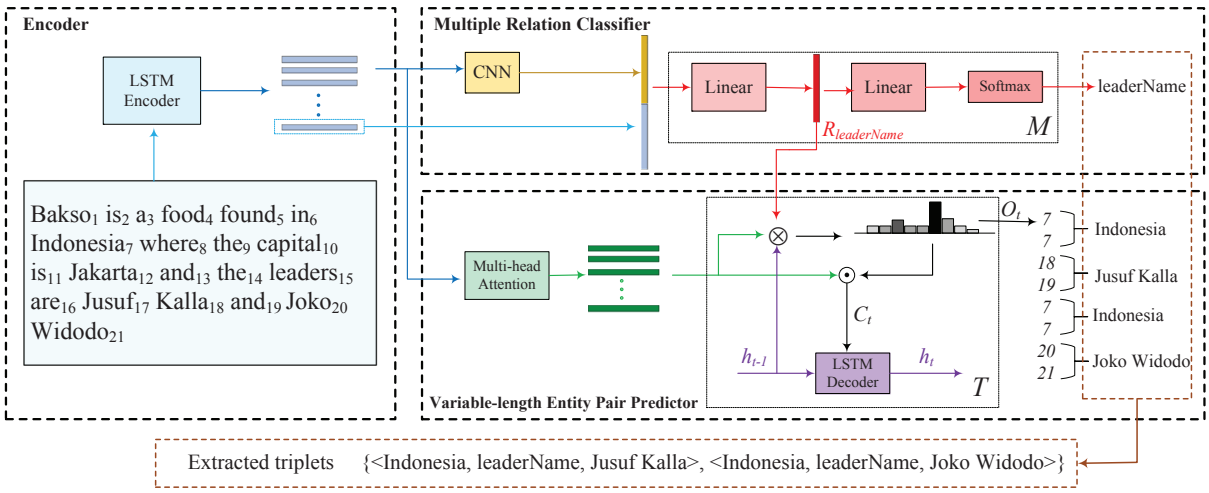


Figure 2: The model architecture of MrMep.

for M relation types, whose output is the probability distribution over whether the corresponding relation is a possible relation or not.

We adopt a convolutional neural network with the same structure as in Kim (2014). Given the text representation $X \in \mathbb{R}^{n \times d}$. A convolution operator and max-over-time pooling (Collobert et al., 2011) operator are applied:

$$C = Conv(X), \tag{2}$$

$$Q = relu(max(C)), \tag{3}$$

where $C \in \mathbb{R}^{m \times (n-l+1)}$ is the feature map output by the convolution operator, m is the number of different filters, n is the length of the text, l is the convolutional filter size. Eq. (3) applies an max-over-time pooling operator on feature map C first, and then a $relu$ activation is used to obtain the text embedding $Q \in \mathbb{R}^m$. When sliding over the text with a window size l, different filters offer a variety of l-gram compositions. Therefore, the text embedding Q is viewed as local feature vector of the text in our model.

In order to make better use of the features extracted by LSTM and CNN, a concatenation operator is used between the last word representation X_n in the encoder and text embedding Q from CNN to produce a fused vector $H \in \mathbb{R}^{m+d}$:

$$H = Concat(Q, X_n). \tag{4}$$

The binary classifier for j-th relation type is shown as follows (We omit the bias b for simplification):

$$R_j = HW_j^H, \tag{5}$$

$$Y^j = softmax(R_j W_j^R). \tag{6}$$

As in Eq. (5), a linear layer is applied to produce the hidden layer state $R_j \in \mathbb{R}^d$. $W_j^H \in \mathbb{R}^{(m+d) \times d}$ is a learnable weight matrix. As in Eq. (6), another linear layer with a $softmax$ activation function is used to predict the probability distribution of whether the text contains the j-th relation type or not, $Y^j \in \mathbb{R}^2$. $W_j^R \in \mathbb{R}^{d \times 2}$ is weight parameter. The hidden layer state R_j is viewed as the relation embedding for j-th relation type.

If the text contains j-th relation type, R_j will be fed into variable-length entity pair predictor to aid entity pair recognition.

3.4 Variable-length Entity Pair Predictor

Given a text, and a target relation type output by the relation classifier, the variable-length entity pair predictor aims to extract its all possible entity pairs in a sequence manner. Inspired by the pointer network (Vinyals et al., 2015), we determine an entity by identifying START and END position indexes of the words in the text. As shown in Figure 2, entity pairs are generated by a sequence of indexes. Each two indexes can identify an entity and each two entities form an entity pair in order. In this paradigm, our model can explore all the possible relations in one pass and predict all possible entity pairs for a given relation via a lightweight sequence decoder, unlike previous work which has to predict the target relation in a multi-pass way (Zeng et al., 2018; Takanobu et al., 2019).

Multi-head Attention. We apply a multi-head attention (Vaswani et al., 2017) on X to obtain a

new text representation P:

$$Q = XW_j^Q, K = XW_j^K, V = XW_j^V, \quad (7)$$

$$head_j = softmax(\frac{QK^T}{\sqrt{d_k}})V, \quad (8)$$

$$P = Concat(head_1, ...head_h)W^O, \quad (9)$$

where different linear layers are used in Eq. (7) to map X into different subspaces by learnable parameters W_j^Q, $W_j^K \in \mathbb{R}^{d \times d_k}$, and $W_j^V \in \mathbb{R}^{d \times d_v}$, yielding query $Q \in \mathbb{R}^{n \times d_k}$, keys $K \in \mathbb{R}^{n \times d_k}$ and values $V \in \mathbb{R}^{n \times d_v}$. By Scaled Dot-Product Attention (Vaswani et al., 2017), $head_j \in \mathbb{R}^{n \times d_v}$ is obtained as the representation of j-th head. A concatenation operator and a linear layer are used in Eq. (9) to produce text representation $P \in \mathbb{R}^{n \times d}$, where h is the number of heads and $W^O \in \mathbb{R}^{hd_v \times d}$ is learnable parameter.

Triplet Attention. At each position of the text, attention mechanism is used to obtain a weighted value that represents the matching degree between the token and the target relation type. Since its aim is to extract the candidate entity pair for the target relation, we call this attention the triplet attention. The triplet attention essentially aggregates the matching of the relation type to each token of the text and used the aggregated matching result to make the entity prediction.

Assume the j-th relation is the target relation identified by the relation classifier in Section 3.3. To get the final attention of j-th relation to i-th token, we study two different modes to implement the triplet attention: paralleled mode and layered mode.

Paralleled Mode. The so-called paralleled mode draws the connections among the target relation, the token, and previous hidden state in a synchronous way:

$$a_t^i = W^a tanh(W^r \circ R_j + W^d \circ d_{t-1} + W^p \circ P_i), \quad (10)$$

where $\circ$ is the element-wise multiplication operator. $R_j \in \mathbb{R}^d$ is j-th relation embedding, $P_i \in \mathbb{R}^d$ is i-th word of text representation P, $d_{t-1} \in \mathbb{R}^d$ is hidden state of LSTM decoder at time step $t-1$ which obtained by Eq. (13). W^r, W^d, W^p, and $W^a \in \mathbb{R}^d$ are learnable parameters. $a_t^i \in \mathbb{R}$ is attention weight of i-th word in the text.

The attention distribution on text $\alpha_t \in \mathbb{R}^n$ is computed as follows:

$$\alpha_t = softmax(a_t), \quad (11)$$

where $\alpha_t = [\alpha_t^1, ...\alpha_t^n]$, n denotes the text length. It is worthy to note that, α_t^i is used as the probability that position index i is selected as output at time step t. At time step t, the position index with the highest probability in α_t is greedily sampled as the output O_t.

Once the attention weights are computed, the context vector c_t is computed by:

$$c_t = \sum_{i=1}^{n} \alpha_t^i \cdot P_i. \quad (12)$$

Then c_t together with d_{t-1} are fed into LSTM decoder at time step t:

$$d_t = LSTM_{decoder}(c_t, d_{t-1}). \quad (13)$$

Therefore, the LSTM decoder is capable of tracking the state of the variable-length entity pair predictor.

Layered Mode. The layered mode first computes the connection between R_j and d_{t-1}:

$$\beta_t = tanh(W^{r'} \circ R_j + W^{d'} \circ d_{t-1}), \quad (14)$$

where W^β, $W^{p'}$, and $W^{a'} \in \mathbb{R}^d$ are learnable parameters.

Then vector β_t is used to calculate triplet attention:

$$a_i^t = W^{a'} tanh(W^\beta \circ \beta_t + W^{p'} \circ P_i), \quad (15)$$

where W^β, $W^{p'} \in \mathbb{R}^d$ are learnable parameters. The following procedure is conducted as the same denoted in Eq. (11)-(13).

3.5 Training

We adopt the cross-entropy loss function to define the loss of multiple relation classifiers and variable-length entity pair predictor respectively, which denoted as L_{rel} and L_{ent}.

To jointly train the model, the loss L is obtained by:

$$L = \lambda \cdot L_{rel} + (1 - \lambda) \cdot L_{ent}, \quad (16)$$

where $\lambda \in \mathbb{R}$ is a hyperparameter to balance the multiple relation classifiers and the variable-length entity pair predictor.

4 Experiments

4.1 Dataset

In order to test our proposed recipe for jointly extracting multiple relations and multiple entity

Dataset	NYT	WebNLG	SKE
Language	English	English	Chinese
Relation	24	246	50
Triplets	104,518	12,863	111,539
Token	90,760	5,051	170,206

Table 1: Statistics of three datasets in language, number of relation types, triplet number and token numbers.

pairs, we conducted experiments on two English benchmark datasets: New York Times (NYT) and WebNLG, and a Chinese public dataset Schema based Knowledge Extraction (SKE).

NYT is produced by distant supervision method (Riedel et al., 2010) and widely used in the triplet extraction (Zheng et al., 2017; Zeng et al., 2018). There are 24 relation types, 61,265 train texts and 5,000 test texts in NYT dataset. We randomly select 5,000 texts from train data as the development set.

WebNLG dataset (Gardent et al., 2017), is originally a dataset for natural language generation task and later used for triplet extraction (Zeng et al., 2018). There are 246 relation types, 5,321 train texts and 695 test texts. We randomly select 500 from train data as the development set.

Different from the two previous English datasets, SKE is a Chinese dataset for information extraction, which is released in the 2019 Language and Intelligence Challenge[1]. SKE contains 50 relation types and training texts exceed 200,000. We build our training set, development set, and test set by randomly selecting 50,000, 5,000 and 5,000 texts respectively.

Following the work of Zeng et al. (2018), we prepropose the three datasets as follows: (1) remove texts that contain no triplets at all; (2) filter out texts if there is an entity in the triplet that is not found in the text.

As shown in Table 1, SKE has a richest vocabulary (170,206 tokens), while WebNLG has a smallest vocabulary (5,051 tokens). In contrast to NYT and SKE which has a medium body of relations (24 and 50, respectively), WebNLG has a significantly big body of relations (246 relations).

4.2 Baseline and Evaluation Metrics

To evaluate our method, we compared against one baseline model and three state-of-the-art models.

[1]http://lic2019.ccf.org.cn/kg

Hyperparameter	value
dropout rate	0.5
learning rate	0.001
batch size	50
hidden size of LSTM	100
filter number of CNN	100
window size of CNN	3
head number	4
λ	0.3

Table 2: Hyperparameter setting.

Baseline: In our proposed baseline model, we design similar architecture with MrMep, in which the encoder and multiple relation classifiers are the same with MrMep, but we use Match-LSTM (Wang and Jiang, 2016) as an implementation of variable-length entity pair predictor. The three main layers of Match-LSTM in our baseline model are as follows: (1) LSTM Preprocessing Layer: we use output of the encoder as passage representation and relation embedding of multiple relation classifiers as the query representation; (2) Match-LSTM Layer: we make concatenation of query representation and each token embedding of passage representation to obtain query-aware token representation, and then fed it into a Bi-LSTM layer; (3) Answer Pointer Layer: this layer is same with Match-LSTM (Wang and Jiang, 2016) and we adopt the sequence model to produce entity pairs.

Tagging (Zheng et al., 2017): This model is a sequence labeling model which assigns to each token a unique tag denoting the information of relation, head or tail entity, even if that the token participates in two different triplets.

CopyR (Zeng et al., 2018): This model is a seq2seq model utilizing copy mechanism that generating a triplet by jointly copying a relation from relation set and copying an entity pair from the source texts in a sequential manner.

HRL (Takanobu et al., 2019): This model applies a hierarchical reinforcement learning framework that decomposes the task into a high-level task for relation detection and a low-level task for entity extraction, which is jointly optimized through a reinforcement learning paradigm.

Following (Zheng et al., 2017; Zeng et al., 2018; Takanobu et al., 2019), we use the standard micro Precision, Recall and F1 score to evaluate the results. Triplets are regarded as correct when the

Model	NYT			WebNLG			SKE		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Tagging	0.526	0.336	0.410	0.476	0.186	0.267	0.347	0.158	0.220
CopyR	0.602	0.556	0.578	0.381	0.369	0.375	0.402	0.362	0.380
HRL	0.741	0.651	0.693	**0.695**	0.629	0.660	0.582	0.422	0.489
Baseline	0.743	0.730	0.736	0.641	0.744	0.689	0.607	0.560	0.583
MrMep (para)	0.769	0.730	0.747	**0.695**	0.759	0.725	0.589	0.543	0.565
MrMep (layer)	**0.779**	**0.766**	**0.771**	0.694	**0.770**	**0.730**	**0.611**	**0.567**	**0.588**

Table 3: Results of different models in three datasets. MrMep (para) denotes that we adopt paralleled mode of triplet attention in variable-length entity pair predictor. MrMep (layer) denotes the layered mode.

Model	NYT10			NYT11			NYT10-sub			NYT11-plus		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Tagging	0.593	0.381	0.464	0.469	0.489	0.479	0.256	0.237	0.246	0.292	0.220	0.250
CopyR	0.569	0.452	0.504	0.347	0.534	0.421	0.392	0.263	0.315	0.329	0.224	0.264
HRL	0.714	0.586	0.644	**0.538**	**0.538**	**0.538**	0.815	0.475	0.600	0.441	0.321	0.372
MrMep	**0.717**	**0.635**	**0.673**	0.434	0.522	0.474	**0.832**	**0.550**	**0.662**	**0.512**	**0.327**	**0.399**

Table 4: Detailed results of different models in datasets used in HRL (Takanobu et al., 2019). NYT10-sub is selected from NYT10 test set (original version), NYT11-plus is splitted from NYT-11 train set (manually created version). NYT10-sub and NYT11-plus contains a variety of overlapping triplets.

relation type and entity pair are both correct.

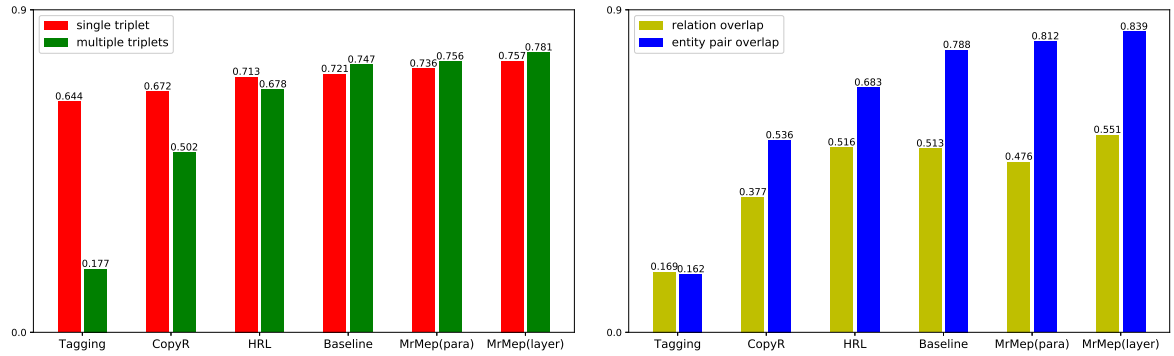
4.3 Implementation Details

We evaluate two variants of our approach: MrPep (para) using paralleled mode for triplet attention and MrMep (layer) using layered mode. For both variants, all hyperparameters are tuned on the same validation set. The hyperparameter setting is shown in Table 2. The word embedding are initialized using Glove (Pennington et al., 2014) and are updated during training, and the dimension is set to 100. The cell unit number of LSTM encoder and decoder is set to 100. The filter number used in CNN classifier is 100, filter window is 3, and the following dense layer has a hidden layer with 100 dimensions. The learning rate is set to 0.001. The tradeoff parameter λ in loss function is set to 0.3. The batch size is set to 50. The head number in muti-head attention is set to 4. For training, we use Adam (Kingma and Ba, 2015) to optimize parameters.

4.4 Main Results

Table 3 shows the Precision, Recall and F1 value of different models on the three datasets. When compared to Tagging and CopyR, the proposed model MrMep substantially improves the F1 scores in three datasets. Both the MrMep (para) and MrMep (layer) outperform the most competitive HRL model in three F1 scores. Moreover, compared with HRL, both the baseline model and our proposed models achieve a significant increase in Recall. Specifically, MrMep (layer) achieves 11.5% improvement in NYT, 14.1% improvement in WebNLG, and 14.5% improvement in SKE in Recall value. Our hypothesis is that the state-of-the-art models such as CopyR and HRL are seeking connection between relation and entity pairs in a sequential fashion, whereas we seek this connection in an interactive manner with the relation paying attention to the tokens one by one interactively.

To draw a paralleled comparison with HRL in more details, we compare the results on the same data partition used in HRL: NYT10, NYT11, NYT10-sub and NYT11-plus. The results of different approaches are summarized in Table 4. On NYT10, NYT10-sub and NYT11-plus test set, MrMep (layer) achieves the highest F1 values when compared with three state-of-the-art models. In NYT11 test set, our model gets poorer performance than HRL. But it is worthy to note that NYT11 dataset averagely contains a triplet per text (totally 369 texts with 370 triplets), while NYT10-sub and NYT11-plus test set contain more overlapping triplets. These observations suggest that our model is essentially feasible to extract more complicated triplets.



(a) (b)

Figure 3: Detailed results on different test subsets. The ordinate of the coordinate axis represents the F1 value.

Model	NYT	WebNLG	SKE
MrMep	**0.771**	**0.730**	**0.588**
w/o CNN	0.745	0.724	0.530
w/o Multi-head	0.723	0.616	0.512

Table 5: An ablation study for MrMep (layer) model. All values are F1 scores.

4.5 Detailed Results

To further verify the ability of MrMep in handling the various complex triplets, we conduct a series of qualitative analysis from two distinct views:

Single triplet vs Multiple triplets. According to the number of triplets contained in the text, we divide the NYT test set into two sub-datasets: (1) single triplet test set: each text contains only one triplet, and 3,240 texts are in the set; (2) multi triplet test set: each text contains two or more triplets, and 1,760 texts are in the set.

Relation overlap vs entity pair overlap. According to whether an overlap is relation overlap or entity pair overlap, we obtain two subsets from the test set: (1) relation overlap: one relation connects with two or more different entity pairs (462 texts in NYT test set); (2) entity pair overlap: one entity pair connects with two or more relations (969 texts in NYT test set).

Figure 3 (a) shows the F1 values of different approaches on NYT test data with single triplet and multiple triplets. It can be seen that both MrMep (para) and MrMep (layer) outperform the baseline and the three comparative models. Noticeably, the improvements on multiple triplets are more remarkable.

From Figure 3 (b) we can observe that MrMep (layer) outperforms MrMep (para) and both of them outperform the reference models. It is also interesting to notice that predicting relation overlaps is more challenging than predicting entity pair overlaps.

The baseline model adopts Match-LSTM (Wang and Jiang, 2016) as an implement of entity pair predictor, which sequentially aggregates the matching of the attention-weighted query to each token of the text. Although we could treat the relation as a query, the relation is essentially a tag with much simpler semantics than that of actual queries. Therefore, we design a lightweight MrMep model which is proven to be more elegant and powerful for entity pair extraction.

4.6 Ablation Study

We examine the contributions of two main components, namely, convolutional neural network (CNN) in multiple relation classifiers and multi-head attention in entity pair predictor, using the best-performing MrMep model with layered mode on three dataset. First, instead of using representations learnt by CNN, we use the last hidden state of the LSTM encoder as text presentation, then feed it directly to the multiple binary classifier (MrMep w/o CNN). Second, instead of applying multi-head attention, we use the hidden state at each time step of the LSTM encoder to represent each token, which is directly fed into the following triplet attention to extract entity pairs (MrMep w/o Multi-head). Table 5 shows the results. We can observe that adding either CNN or multi-head attention improves the performance of the model. This suggests that both parts can assist MrMep to jointly extract entities and relations, where the CNN layer seems to be playing a more significant role. One possible reason is that CNN is used for

MrMep (layer)	(NYT): "It 's pretty neat", said Ward, who grew up in the [Edmonton]$_{E_t-contains}$ suburb of Sherwood Park, [Alberta]$_{E_h-contains}$.
MrMep (para)	(NYT): "It 's pretty neat", said Ward, who grew up in the [Edmonton suburb of Sherwood Park]$_{E_t-contains}$, [Alberta]$_{E_h-contains}$.
MrMep (layer)	(WebNLG): [Bakso]$_{E_h-country}$ comes from [Indonesia]$_{E_t-country}$, and celery is a main ingredient. [Jasuf Kalla]$_{E_t-leaderName}$ is a leader in [Indonesia]$_{E_h-leaderName}$.
MrMep (para)	(WebNLG): [Bakso]$_{E_h-country}$ comes from [Indoneisa]$_{E_t-country}$, and [celery is a main ingredient. Jasuf Kalla]$_{E_t-leaderName}$ is a leader in [Indonesia]$_{E_h-leaderName}$.

Table 6: Extracted results from MrMep (para) and MrMep (layer). The words in a bracket represents an entity, E_h stands for the head entity and E_t stands for the tail entity. Red is marked as identifying the wrong.

extracting local feature of input text, which is not only beneficial for relation extraction but also assist producing better relation embedding to aid entity pair predition.

4.7 Case Study

Table 6 shows two examples of extracted results from MrMep (para) and MrMep (layer). In the first example, MrMep (layer) precisely extracts the correct triplet <Alberta, contains, Edmonton>, while MrMep (para) recognizes incorrect tail entity " Edmonton suburb of Sherwood Park". In the second example, MrMep (layer) precisely extracts correct triplet <Indoneisa, leaderName, Jasuf Kalla> while MrMep (para) wrongly recognizes the tail entity " celery is a main ingredient. Jasuf Kalla". Taking NYT test set as example, among 1250 differences between the output of MrMep (para) and MrMep (layer), there are 885 differences resulting from the wrong entity boundaries identified by MrMep (para). The most likely reason might be that the hidden state of the decoder plays as crucial role as the relation in sequentially producing different entity pairs.

5 Conclusion

We propose a joint multiple relation and multiple entity pair extraction model. The model uses a triplet attention to model the connection of relation and entities in a novel and lightweight framework. It gives state-of-the-art performance on three benchmark datasets.

References

Yee Seng Chan and Dan Roth. 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*, pages 551–560. Association for Computational Linguistics.

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

Rajarshi Das, Tsendsuren Munkhdalai, Xingdi Yuan, Adam Trischler, and Andrew McCallum. 2019. Building dynamic knowledge graphs from text using machine reading comprehension. *In Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 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)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2537–2547.

Shizhu He, Cao Liu, Kang Liu, and Jun Zhao. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 199–208.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *Proceedings of ICLR*, pages 1–15.

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.

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

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)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

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.

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. Springer.

Benjamin Roth, Costanza Conforti, Nina Poerner, Sanjeev Kumar Karn, and Hinrich Schütze. 2019. Neural architectures for open-type relation argument extraction. *Natural Language Engineering*, 25(2):219–238.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ICLR 2017*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

Xiaofeng Yu and Wai Lam. 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*, pages 1399–1407. Association for Computational Linguistics.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106.

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)*, pages 506–514.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1730–1740.

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)*, pages 1227–1236, Vancouver, Canada. Association for Computational Linguistics.