Final Report for Multi-Sentence Relation Extraction

Lixin Huang, Xingcheng Yao

Abstract

A document generally mentions many entities exhibiting complex cross-sentence relations. Most existing methods typically focus on inner-sentence relation extraction and thus are inadequate to collectively identify these relational facts from a long document. To address the challenging task of multi-sentence relation extraction, we propose a novel framework with (1) a knowledge memory module to record the useful knowledge about entities and semantics of sentences during reading the document sentence by sentence, and (2) a relational reasoning module to jointly infer cross-sentence entity relations over the knowledge memory. Experimental results show that our models scale well to long documents with numerous sentences and significantly outperform the baseline models.

1 Introduction

Relation extraction (RE) aims to automatically identify relational facts between entities scattered in open-domain text, which is an active research area and essential to the development of large-scale knowledge graphs (KGs). Most works on RE devote to extracting the relation of two entities mentioned within one sentence. In recent years, with the rapid development of neural networks, various deep models have been explored to encode relational patterns of two entities from a sentence for RE and achieve the state-of-the-art performance.

Besides those relational facts of inner-sentence entity pairs, more relational facts exist among entities scattered in multiple sentences of a document. Hence, we argue to move RE forward from the inter-sentence level to the multi-sentence level and further research how to handle two key challenges for multi-sentence relation extraction: (1) Given a long document consisting of multiple sentences, there are rich semantic and knowledge information with long-term dependencies. It is essential for a multi-sentence RE system to have the **knowledge memory** function to memorize these long-term information about entities so as to extract their relations. (2) Given many entities mentioned in a document, the relations between these entities exhibit complex relationships with each other. Hence, multi-sentence RE also requires the **relational reasoning** function for inferring new facts according to some basic facts.

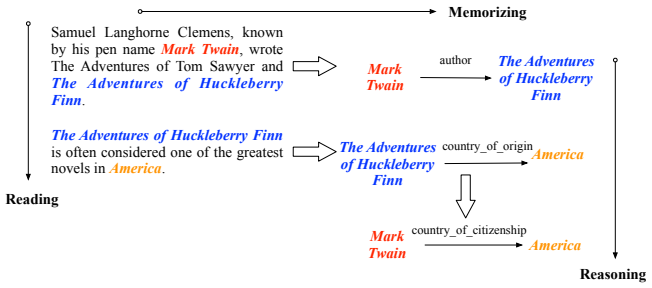

Figure 1: An example of reasoning over different sentences in a document together for relation extraction.

Taking Figure 1 for example, multi-sentence RE is expected to first detect and memorize (*The Adventures of Huckleberry Finn*, `author`, *Mark Twain*) and (*The Adventures of Huckleberry Finn*, `country_of_origin`, *America*) by reading all the sentences in the document, and then reason over these relations to identify the new fact (*Mark Twain*, `country_of_citizenship`, *America*).

Some pioneering works have been explored for multi-sentence RE [Wick *et al.*, 2006; Gerber and Chai, 2010; Swampillai and Stevenson, 2011; Yoshikawa *et al.*, 2011; Quirk and Poon, 2017]. These methods typically rely on lexical and syntactic patterns as textual features for relation classification, which inevitably accompany with data sparsity and limit the capacity of memorizing and reasoning. Some works try to improve the memorizing ability by applying sophisticated recurrent neural networks such as graph LSTM [Peng *et al.*, 2017; Song *et al.*, 2018], however, their packing all the history information into one hidden state vector potentially forces reasoning less tractable. Moreover, these works only extract the relation of two specific entities from a document, and less work has been done to collectively extract complex relations among multiple entities simultaneously.

As shown in Figure 2, we propose a novel framework for multi-sentence RE with enhanced memorizing and reasoning abilities by leveraging a knowledge memory module and a relational reasoning module, which are introduced as follows.

Knowledge Memory. To better distinguish different kinds of information for multi-sentence RE, the knowledge mem-

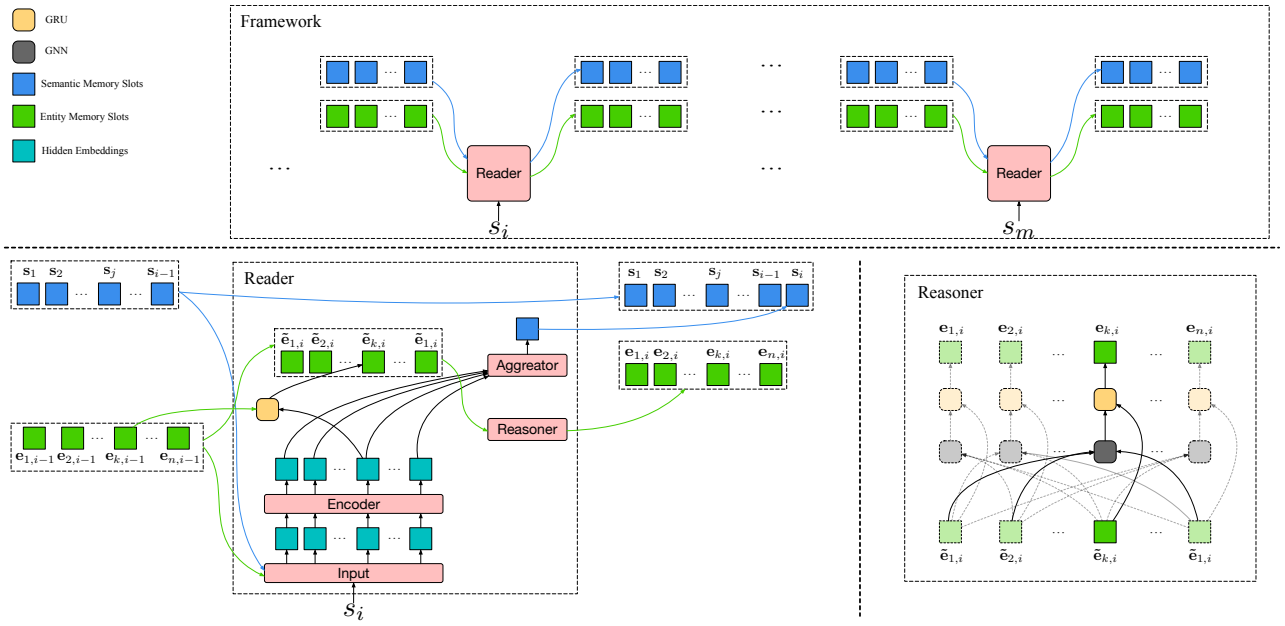

Figure 2: **The framework of our model.** Each slot of the semantic memory and entity memory is corresponding to a sentence and an entity respectively. The useful knowledge about the entities and sentences will be gradually gathered into the knowledge memory while the document $\mathcal{D} = \{s_1, \ldots, s_m\}$ is read sentence by sentence. The components in the reader indicates the execution order when encoding a sentence s_i. The components in the reasoner show the details for each slot of the entity memory when performing reasoning.

ory module is designed to consist of two parts: the semantic memory to memorize the semantic meanings of sentences, and the entity memory to store the knowledge about entities. Each slot of the two parts is corresponding to a sentence or an entity respectively. When reading the document sentence by sentence, the semantic information of preceding sentences and the entity knowledge of known entities will be gathered in order to understand the following sentences.

Relational Reasoning. To infer the implicit relations between entities implied by the history context, we perform relational reasoning over the knowledge memory each time after reading each sentence, and the reasoning results for entities will be updated into their corresponding slots of the entity memory. In this way, the model is capable of inducing the cross-sentence interactions of entities, and supports collective identification of complex relations among these entities. Moreover, by updating the knowledge memory with the reasoning results periodically, it can also help the reading of the following sentences more effectively.

In fact, to address the memorizing and reasoning issues in various tasks such as question answering and block puzzle game, memory augmented neural networks have been proposed with promising results [Weston *et al.*, 2014; Graves *et al.*, 2014; Sukhbaatar *et al.*, 2015; Graves *et al.*, 2016; Santoro *et al.*, 2018]. Among these works, [Santoro *et al.*, 2018] achieves the state-of-the-art performance by proposing relational memory core (RMC). However, these models generally design memories as a set of multiple embeddings, with limited discriminability in multi-sentence RE as will shown in our experiments. In contrast, we design knowledge memory in our model as entity-wise and sentence-wise, which can better support memorizing and reasoning for multi-sentence RE.

For experiments, we test our proposed model on a large-scale dataset WikiDRE, and the experimental results show that the proposed memorizing and reasoning schemes significantly outperform other baseline methods, including the recent state-of-the-art models, empirically demonstrating the essentiality and effectiveness of memorizing and reasoning abilities for multi-sentence RE.

2 Related Work

2.1 Relation Extraction

Neural network architectures are widely used in RE and focus on extracting inner-sentence relations, including convolutional neural networks [Liu *et al.*, 2013; Zeng *et al.*, 2014; Santos *et al.*, 2015], recurrent neural networks [Zhang and Wang, 2015; Vu *et al.*, 2016; Zhang *et al.*, 2015; Zhou *et al.*, 2016; Xiao and Liu, 2016], dependency-based neural models [Socher *et al.*, 2012; Liu *et al.*, 2015; Cai *et al.*, 2016], and bag-level models [Zeng *et al.*, 2015; Lin *et al.*, 2016; Wu *et al.*, 2017; Qin *et al.*, 2018].

Some works have also devoted to extracting relations cross multiple sentences in a document, which cannot be handled by the above methods designed for inner-sentence RE. The early methods [Wick *et al.*, 2006; Gerber and Chai, 2010; Swampillai and Stevenson, 2011; Yoshikawa *et al.*, 2011; Quirk and Poon, 2017] rely on textual features extracted from various dependency structures, such as co-reference annota-

tions, parse trees and discourse relations, without considering the memorizing and reasoning abilities. Then, Peng *et al.* [2017] and Song *et al.* [2018] employ graph-structured recurrent neural networks to model cross-sentence dependencies for RE, which have limited reasoning and memorizing abilities. Moreover, these cross-sentence methods only utilize documents to identify the relation of a specific entity pair each time. In this work, we propose a model that collectively identifies all relational facts of multiple entities in multi-sentence documents, which is a more challenging task that requires reading, memorizing, and reasoning for discovering relational facts from multiple sentences.

2.2 Memory Augmented Neural Networks

As the rapid development on memory augmented neural networks, these memory models provide an effective approach to supporting memorizing and reasoning for long sequential data. One of the earliest methods with a memory component is Memory Networks [Weston *et al.*, 2014], whose memory is built from inputs and it reads via a sophisticated attention-based addressing mechanism. Unfortunately, it requires heavy supervision of which memory slots to attend in training. The successor End-To-End Memory Network (MemN2N) [Sukhbaatar *et al.*, 2015] alleviates the drawback by employing a simpler addressing mechanism. Neural Turing Machine (NTM) [Graves *et al.*, 2014] and Differentiable Neural Computer (DNC) [Graves *et al.*, 2016] are similar to Memory Networks. They add a write operation to update the memories following the read operation.

The memories of all the above models lack the mechanism to interact internally, and struggle to resolve those relational reasoning tasks which involve strong entity interactions [Santoro *et al.*, 2018]. Relational Memory Core (RMC) [Santoro *et al.*, 2018], the most relevant work to us, alleviates the problem by employing multi-head dot product attention to allow memories to interact, and achieves promising results on various relational reasoning tasks. As compared to RMC, our model is specially designed for multi-sentence RE, and shows the following advantages: (1) We divide the knowledge memory into two parts, semantic memory and entity memory, with better discriminability for modeling the history information while reading. (2) We set a memory slot for each entity explicitly, and can flexibly model their interactions within the memory. (3) With the entity-wise and sentence-wise architecture, we can perform updating and reasoning over the knowledge memory sentence by sentence, which is more computationally efficient than RMC. In experiments, we empirically compare these memory models and demonstrate the effectiveness of our model for multi-sentence RE.

3 Methodology

In this section, we will introduce the overall framework of our model which reasons over knowledge memory to understand long sequential data for RE.

3.1 Notations

We denote a document consisting of multiple sentences as $\mathcal{D} = \{s_1, \ldots, s_m\}$, where each sentence $s_i \in \mathcal{D}$ consists of several words $s_i = \{w_{i,1}, \ldots, w_{i,|s_i|}\}$. There are also some named entities mentioned in some sentences of a document $\mathcal{D}$, referred as $\mathcal{E}_\mathcal{D} = \{e_1, \ldots, e_n\}$.

In this work, we adopt a semantic memory $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$ for $\{s_1, \ldots, s_m\}$, where $\boldsymbol{s}_i$ stores sentence features for s_i. In addition, we adopt an entity memory $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ to store entity features for $\{e_1, \ldots, e_n\}$. The intuition behind this approach is that in order to better grasp the relationship between the two entities, when reading a sentence in a document, we not only need to extract the general information provided by this sentence under the context, but also need to focus on information related to the entities. The former ensures that we do not misunderstand the overall meaning of the document, and the latter ensures that there is not too much entity-independent noise in the extracted information.

Because we sequentially encode each sentence in the document and update the memories, we denote $\boldsymbol{e}_{k,i}$ as the entity memory $\boldsymbol{e}_k$ after encoding the sentence from s_1 to s_i.

3.2 Framework

Given several entities $\mathcal{E}_\mathcal{D} = \{e_1, \ldots, e_n\}$ in a document $\mathcal{D}$, we adopt our model to measure the probability of each relation $r \in \mathcal{R}$ (including a special relation "NA" indicating the relation between an entity pair is not available) holding between **any two of these entities**. As shown in Figure 2, we encode $\mathcal{D}$ sentence by sentence, and the overall framework includes four core components: (1) a semantic memory for storing sentence information, (2) an entity memory for storing entity information, (3) a reasoning module for reasoning and synthesizing information over the entity memory, and (4) a sentence reader with word embeddings, position embeddings and memory embeddings as input for encoding sentences and then updating memory modules.

To be specific, given a sentence $s_i = \{w_{i,1} \ldots, w_{i,|s_i|}\}$, the sentence reader first uses word and position embedding [Zeng *et al.*, 2014] for each word $w_{i,j}$ to compute its input embedding $\boldsymbol{x}^I_{i,j}$,

$$\boldsymbol{x}^I_{i,j} = \boldsymbol{w}_{i,j} + \boldsymbol{p}_{i,j}, \tag{1}$$

where $\boldsymbol{w}_{i,j}$ and $\boldsymbol{p}_{i,j}$ are word embedding and position embedding respectively. Then, we use the input embedding $\boldsymbol{x}^I_{i,j}$ to gather information correlated with this word both from the semantic and entity memories,

$$\begin{aligned} \boldsymbol{x}^S_{i,j} &= \texttt{S-MEM}(\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{i-1}\}, \boldsymbol{x}^I_{i,j}), \\ \boldsymbol{x}^K_{i,j} &= \texttt{E-MEM}(\{\boldsymbol{e}_{1,i-1}, \ldots, \boldsymbol{e}_{n,i-1}\}, w_{i,j}), \\ \boldsymbol{x}_{i,j} &= \boldsymbol{x}^I_{i,j} + \boldsymbol{x}^S_{i,j} + \boldsymbol{x}^K_{i,j}, \end{aligned} \tag{2}$$

where $\texttt{S-MEM}(\cdot,\cdot)$ and $\texttt{E-MEM}(\cdot,\cdot)$ are defined as the function to extract information from semantic memory and entity memory respectively, which will be illustrated in detail in Section 3.4. Based on the sequential features $\{\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|s_i|}\}$, an encoding layer of the sentence reader is applied to obtain the hidden embeddings of all words,

$$\{\boldsymbol{h}_{i,1}, \ldots, \boldsymbol{h}_{i,|s_i|}\} = \texttt{Encoder}(\{\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|s_i|}\}), \tag{3}$$

where $\texttt{Encoder}(\cdot)$ is the neural encoding layer.

As soon as finishing encoding the sentence s_i, the hidden embeddings of the encoding layer will be updated into the semantic and entity memories. For the **semantic memory**, the hidden embeddings are aggregated into a united sentence representation which will be stored into $\boldsymbol{s}_i$,

$$\boldsymbol{s}_i = \texttt{Aggregator}(\{\boldsymbol{h}_{i,1}, \ldots, \boldsymbol{h}_{i,|s_i|}\}), \tag{4}$$

where $\texttt{Aggregator}(\cdot)$ is the neural operation to compute the sentence representation. The details of $\texttt{Encoder}(\cdot)$ and $\texttt{Aggregator}(\cdot)$ will be illustrated in Section 3.3.

While for the **entity memory**, if an entity e_k is corresponding to the word $w_{i,j}$ in the sentence s_i, $\boldsymbol{h}_{i,j}$ will be updated into the entity memory through a gated recurrent unit (GRU) [Cho *et al.*, 2014].

$$\tilde{\boldsymbol{e}}_{k,i} = \texttt{GRU}(\boldsymbol{e}_{k,i-1}, \boldsymbol{h}_{i,j}), \tag{5}$$

where $\tilde{\boldsymbol{e}}_{k,i}$ is the intermediate entity memory after updating the sentence s_i into the entity memory. For other entities which are not mentioned in the sentence s_i, their memory features stay the same as before: $\tilde{\boldsymbol{e}}_{k,i} = \boldsymbol{e}_{k,i-1}$.

After updating the memory with the sentence s_i and before encoding the next sentence s_{i+1}, we treat the entity memory as a fully-connected entity graph, and adopt a reasoning module to propagate information among entities,

$$\{\boldsymbol{e}_{1,i}, \ldots, \boldsymbol{e}_{n,i}\} = \texttt{Reasoner}(\{\tilde{\boldsymbol{e}}_{1,i}, \ldots, \tilde{\boldsymbol{e}}_{n,i}\}). \tag{6}$$

The reasoning module $\texttt{Reasoner}(\cdot)$ will be further explained in detail in Section 3.5. After achieving the semantic memory $\boldsymbol{s}_i$ and the entity memory $\{\boldsymbol{e}_{1,i}, \ldots, \boldsymbol{e}_{n,i}\}$, we will utilize the memory features for encoding the next sentence s_{i+1} and repeat the processing from Eq. (1) to Eq. (6).

Relation of each entity pair will be predicted after the whole document is encoded. For any entity pair $e_i, e_j \in \{e_1, \ldots, e_n\}$, we measure the probability of each relation $r \in \mathcal{R}$ holding between the pair as follows,

$$\begin{aligned} \boldsymbol{r}_{i,j} &= \texttt{Bilinear}(\boldsymbol{e}_{i,m}, \boldsymbol{e}_{j,m}), \\ \boldsymbol{o} &= \boldsymbol{M}\, \boldsymbol{r}_{i,j} + \boldsymbol{b}, \\ P(r|e_i, e_j, \mathcal{D}) &= \frac{\exp(\boldsymbol{o}_r)}{\sum_{\tilde{r} \in \mathcal{R}} \exp(\boldsymbol{o}_{\tilde{r}})}, \end{aligned} \tag{7}$$

where $\boldsymbol{o}$ are the scores of all relations, $\boldsymbol{M}$ and $\boldsymbol{b}$ are the representation matrix and bias vector to calculate the relation scores, $\texttt{Bilinear}(\cdot)$ is a bilinear layer, and $\boldsymbol{e}_{i,m}$ and $\boldsymbol{e}_{j,m}$ are the entity memory features after encoding all sentences. And the loss function is defined as follows,

$$J(\theta) = -\sum_{\mathcal{D}} \sum_{e_i, e_j \in \mathcal{E}_{\mathcal{D}}} \log P(r_{e_i,e_j}|e_i, e_j, \mathcal{D}) + \lambda \|\theta\|_2^2, \tag{8}$$

where r_{e_i,e_j} is the labeled relation for the entity pair $e_i, e_j \in \mathcal{E}_{\mathcal{D}}$, λ is a harmonic factor, and $\|\theta\|_2^2$ is the L2 regularizer.

3.3 Sentence Reader

Given a sentence $s_i = \{w_{i,1}, \ldots, w_{i,|s_i|}\}$ in $\mathcal{D}$, we apply several neural architectures in the sentence reader to get hidden embeddings $\boldsymbol{h}_i$ for capturing semantic and entity information in the sentence.

Input Layer

The input layer of the sentence reader aims to embed both semantic information and positional information of words into their input embeddings which are denoted as $\boldsymbol{x}^I_{i,j}$. For word embeddings, we adopt GloVe [Pennington *et al.*, 2014] to compute $\{\boldsymbol{w}_{i,1}, \ldots, \boldsymbol{w}_{i,|s_i|}\}$. Since our model deals with several entities and each entity may appear in the document for multiple times, we assign each word $w_{i,j}$ an position identification $p_{i,j}$ as follows,

$$p_{i,j} = \begin{cases} k, & w_{i,j} \text{ is corresponding to } e_k \in \mathcal{E}_{\mathcal{D}}, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

Each position identification is represented by a vector $\boldsymbol{p}_{i,j}$. With $\boldsymbol{w}_{i,j}$ and $\boldsymbol{p}_{i,j}$, we can compute the input embedding $\boldsymbol{x}^I_{i,j}$ via Eq. (1), and then gather information from the memories to compute $\boldsymbol{x}_{i,j}$ via Eq. (2).

Encoding Layer

The encoding layer aims to compose $\{\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|s_i|}\}$ into their corresponding hidden embeddings $\{\boldsymbol{h}_{i,1}, \ldots, \boldsymbol{h}_{i,|s_i|}\}$, which acts as $\texttt{Encoder}(\cdot)$ in Eq. (3). In this work, we select two types of neural network architectures, unidirectional and bidirectional LSTM [Hochreiter and Schmidhuber, 1997], and Transformer [Vaswani *et al.*, 2017] to encode sentences. Note that, our framework is independent to the selection of the encoding layers, and it thus can be easily adapted to fit other encoder architectures. In this work, we do not introduce these architectures in detail, and more information can be found from their original papers.

Aggregating Layer

After encoding the sentence s_i and obtaining the hidden embeddings, we will aggregate all the hidden embeddings into united sentence features and store the sentence features into the semantic memory, which acts as $\texttt{Aggregator}(\cdot)$ in Eq. (4). In this work, for unidirectional LSTM and bidirectional LSTM, we design the aggregating layer as selecting the last timestep hidden state vector,

$$\boldsymbol{s}_i = \boldsymbol{h}_{i,|s_i|}. \tag{10}$$

For Transformer, we design the aggregating layer as a max-pooling operation,

$$[\boldsymbol{s}_i]_k = \max_{1 \le j \le |s_i|} [\boldsymbol{h}_{i,j}]_k, \tag{11}$$

where $[\cdot]_k$ is the k-th value of a vector.

3.4 Gathering Information from Semantic and Entity Memories

For encoding each sentence in the document, our model requires to gather information from both the semantic and entity memories storing the preceding sentence and entity features. We design a special attention layer for gathering information from the semantic memory, which acts as $\texttt{S-MEM}(\cdot, \cdot)$

in Eq. (2),

$$e_k = \frac{(\boldsymbol{H}_Q \boldsymbol{x}^I_{i,j}) \cdot (\boldsymbol{H}_K \boldsymbol{s}_k)}{\sqrt{d_h}},$$
$$\alpha_k = \frac{\exp(e_k)}{\sum_{l=1}^{i-1} \exp(e_l)},$$
$$\boldsymbol{x}^S_{i,j} = \texttt{S-MEM}\left(\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{i-1}\}, \boldsymbol{x}^I_{i,j}\right) = \sum_{k=1}^{i-1} \alpha_k \cdot (\boldsymbol{H}_V \boldsymbol{s}_k), \tag{12}$$

where $\boldsymbol{H}_Q$, $\boldsymbol{H}_K$, and $\boldsymbol{H}_V$ are linear transformation matrices. Here $\texttt{S-MEM}(\cdot,\cdot)$ uses the input embeddings $\boldsymbol{x}^I_{i,j}$ as the query vector to perform an attention operation with $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{i-1}\}$ as the key and value vectors, following the attention method proposed by Vaswani *et al.* [2017].

For each word $w_{i,j}$ in the sentence s_i, we use its position identification to gather information from the entity memory, which acts as $\texttt{E-MEM}(\cdot,\cdot)$ in Eq. (2),

$$\boldsymbol{x}^K_{i,j} = \texttt{E-MEM}(\{\boldsymbol{e}_{1,i-1}, \ldots, \boldsymbol{e}_{n,i-1}\}, w_{i,j}) = \begin{cases} \boldsymbol{H}_E \boldsymbol{e}_{k,i-1}, & p_{i,j} = k, \\ \boldsymbol{H}_E \boldsymbol{0}, & p_{i,j} = 0, \end{cases} \tag{13}$$

where $\boldsymbol{0}$ is a padding vector and $\boldsymbol{H}_E$ is a linear transformation matrix. By computing $\boldsymbol{x}^S_{i,j}$ and $\boldsymbol{x}^K_{i,j}$, we finally sum up the information gathered from the memories together with the input embedding $\boldsymbol{x}^I_{i,j}$ as Eq. (2).

3.5 Reasoning over Entity Memory

After updating the information into the entity memory with Eq. (5), we apply reasoning over entities. To be specific, we treat the entity memory as a fully-connected entity graph, and adopt graph neural networks (GNN) to propagate information among entities, which acts as $\texttt{Reasoner}(\cdot)$ in Eq. (6),

$$\boldsymbol{e}_{k,i} = \texttt{GRU}\Big(\sum_{e_j \in \mathcal{N}_{e_k}} \texttt{ReLU}\big(\boldsymbol{W}\tilde{\boldsymbol{e}}_{j,i} + \boldsymbol{b}\big), \tilde{\boldsymbol{e}}_{k,i}\Big), \tag{14}$$

where $\mathcal{N}_{e_k}$ represents the neighbors of the entity e_k in the entity graph. With the above reasoning operations, we can reason over the entity memory to understand entity information in different sentences through a long document. We believe that storing while reasoning is an intuitive method to process information even for humans, which benefits extracting information from long sequential data.

4 Experiments

4.1 Datasets

To test the performance of our model, we utilize a large-scale dataset named WikiDRE for multi-sentence RE. For each sample, a Wikipedia [1] document and all the entities mentioned are given, and a model is required to predict all the relations among all these entities. WikiDRE is constructed in a distant supervision way: all the named entity mentions in a Wikipedia document are identified using the named entity recognition toolkit spaCy [2]. Then the entity mentions are linked to the items in the Wikidata knowledge base (KB) [3]. And the entity mentions corresponding to the same KB IDs are merged. Finally, for each pair of entities e_1 and e_2 mentioned in the document, if there is a Wikidata statement (e_1, e_2, r) stating that the relation r holds between e_1 and e_2, then r is considered to also hold between e_1 and e_2 given the document, otherwise, the special relation "NA" is assigned. To encourage entity interactions, documents too short or with too few entities/relations are discarded. $48,450$ multi-sentence documents are collected in this dataset with distantly supervised labels, we randomly divided them into training, development and test sets with $44,602$, $2,348$ and $1,500$ documents respFectively.

4.2 Baselines

We compare our models with two sets of baselines: (1) Four widely used neural models designed for inner-sentence RE, including CNN-S [Zeng *et al.*, 2014], Transformer-S [Vaswani *et al.*, 2017], LSTM-S [Xu *et al.*, 2015] and BiLSTM-S [Zhang *et al.*, 2015]. For multi-sentence RE, we concatenate all the sentences in a document to form a pseudo sentence and apply these methods to the pseudo sentence. (2) Three neural models with the ability of leveraging cross-sentence information, including ContextAtt [Sorokin and Gurevych, 2017], MEM [Madotto *et al.*, 2018] and RMC [Santoro *et al.*, 2018]. ContextAtt is designed to improve inner-sentence RE by considering context relations, while MEM and RMC are two memory augmented networks which have been shown effective for utilizing history and knowledge and performing relational reasoning respectively.

4.3 Training Details

Adam [Kingma and Ba, 2014] is used to train the models, with initial learning rate 0.001 and batch size 32. The word embeddings are initialized with the 50-dimensional GloVe vectors [4] and jointly trained. The 3-dimensional position embeddings are randomly initialized. For the encoding layer (Section 3.3), the hidden sizes of the unidirectional LSTM, bidirectional LSTM and Transformer are all 256. The layer number is set to 2 for LSTM and 3 for Transformer. 8 attention heads are used for Transformer. Dropout with drop rate 0.5 is applied to each LSTM and Transformer layer.

4.4 Results

Following previous works, AUC is used as the evaluation metric, and the results are shown in Table 1, where RK-NN is our model and the name in the brackets refers to the architecture used as encoding layer (Section 3.3).

Our model with LSTM and BiLSTM as encoding layer outperforms all the baselines with large margins and achieves higher and comparable performance with baselines when Transformer is used, demonstrating the effectiveness of our model. Surprisingly, although BiLSTM-S is designed for inner-sentence RE, it achieves remarkable high performance. Meanwhile, our model with BiLSTM as encoding layer also achieves the overall best result. Thus, we believe BiLSTM is

[1] https://www.wikipedia.org

[2] https://spacy.io/

[3] https://www.wikidata.org/wiki/Wikidata:Main_Page

[4] http://nlp.stanford.edu/data/glove.6B.zip

Model	F1	AUC	Ign F1	Ign AUC
CNN-S [Zeng *et al.*, 2014]	0.513	0.482	0.359	0.240
Transformer-S [Vaswani *et al.*, 2017]	0.546	0.510	0.407	0.275
LSTM-S [Xu *et al.*, 2015]	0.541	0.516	0.398	0.279
BiLSTM-S [Zhang *et al.*, 2015]	0.565	0.528	0.420	0.290
ContextAtt [Sorokin and Gurevych, 2017]	0.549	0.542	0.418	0.283
MEM [Madotto *et al.*, 2018]	0.568	0.535	0.425	0.287
RMC [Santoro *et al.*, 2018]	0.572	0.547	0.413	0.285
RK-NN (Transformer)	0.577	0.536	0.426	0.288
RK-NN (LSTM)	0.585	**0.581**	0.432	0.306
RK-NN (BiLSTM)	**0.592**	0.579	**0.447**	**0.316**

Table 1: Evaluation results on WikiDRE. RK-NN is our model, and the name in the brackets denotes the encoding layer architecture.

Model	LSTM	BiLSTM
RK-NN	**0.306**	**0.316**
RK-NN (−R)	0.291	0.312
RK-NN (−R, −S-MEM)	0.290	0.305
RK-NN (−R, −E-MEM)	0.287	0.294
RK-NN (−R, −S-MEM, −E-MEM)	0.279	0.290

Table 2: Effect of the reasoning module and two memories. "−" denotes removing the corresponding component from the model.

a better architecture for encoding sentences in multi-sentence RE.

As context relation information is explicitly considered in ContextAtt, MEM, RMC and our model, they generally achieve better performance than the other baselines designed for inner-sentence RE, indicating the reasoning and memorizing abilities are essential for multi-sentence RE. Furthermore, we also believe that the significant improvement achieved by our model over ContextAtt, MEM and RMC comes from the better reasoning and memorizing abilities of our model.

To further investigate the contributions of the reasoning and memorizing abilities of our model, ablation experiments are conducted and the results are shown in Table 2, where −R, −S-MEM and −E-MEM indicate removing the reasoning module, the semantic memory and the entity memory respectively. Performance drops remarkably when the reasoning module is removed and drops further if any of the two memories is also removed, justifying both the reasoning and memorizing abilities are essential. Furthermore, removing both of the memories causes larger performance drop than removing only one, indicating that two memories play complementary roles and justifying the advantage of explicitly distinguishing memories for storing different types of information.

Performance on Harder Dataset

Although our model achieves promising results, a nature question is whether its performance will drop dramatically if the dataset becomes harder. Therefore, we investigate the performance of our model on datasets requiring different levels of memorizing and reasoning. Intuitively, the task difficulty generally grows with the number of entities mentioned in a document because the interactions between the entities become more complicated. Thus, we sort the test set in descending order according to the number of entity mentions and show the performance on the first n percent of the sorted test set in Figure 3. We can observe that the performance of

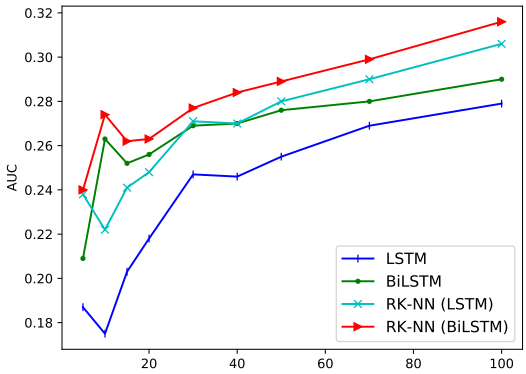

Figure 3: The performance of models on the first n percent of the descendingly sorted WikiDRE test set according to the number of entity mentions.

Model	Time	AUC
LSTM-S [Xu *et al.*, 2015]	5056	0.279
MEM [Madotto *et al.*, 2018]	5112	0.287
RMC [Santoro *et al.*, 2018]	12053	0.285
RK-NN (LSTM)	5185	0.306
RK-NN (BiLSTM)	5399	0.316

Table 3: Training time (s) and AUC of the models.

our model drops slowly as the dataset becomes harder (i.e., n becomes smaller) and its performance on the most difficult 5% samples is even comparable with that of CNN-S on the *entire* test set (Table 1). Therefore, we can conclude that our model is robust. Moreover, our model with LSTM/BiLSTM as encoding layer outperforms LSTM-S/BiLSTM-S consistently with large margins, further justifying the robustness and effectiveness of our model.

4.5 Computational Efficiency

Table 3 shows the training time for one epoch of our model and the baselines (recorded on a Nvidia 2080Ti). Although the architecture of our model with LSTM/BiLSTM as encoding layer is more complex than LSTM-S/BiLSTM-S, their speed is comparable. The memory augmented network MEM also achieves comparable speed. RMC is the most relevant work to ours but both its speed and AUC are significantly lower than ours. Therefore, we conclude that our model is both computationally efficient and effective.

5 Conclusion and Future Work

In this work, we investigate multi-sentence RE which aims to extract all relational facts among multiple entities mentioned in a document, and empirically justify that the memorizing and reasoning abilities are essential for the task. In order to improve these abilities, we propose a novel framework with a knowledge memory module to store entity and sentence information and a relational reasoning module to infer complex entity relations over the memory. Experimental results on the large-scale dataset WikiDRE show the efficiency and

effectiveness of our model as compared to other baselines for multi-sentence RE.

There are a number of interesting directions we would like to pursue in the future: (1) There is rich external knowledge on the Web, which is potentially helpful for multi-sentence RE. Due to the entity-wise architecture of the knowledge memory, our model should be capable of incorporating the external knowledge efficiently, which can be explored in the future. (2) We will investigate more effective methods in graph neural networks for reasoning over the knowledge memory. (3) The dataset WikiDRE is built with distant supervision with inevitable noisy annotations. In the future, we will build a large-scale human-annotated dataset to better evaluate multi-sentence RE.

References

[Cai *et al.*, 2016] Rui Cai, Xiaodong Zhang, et al. Bidirectional recurrent convolutional neural network for relation classification. In *Proceedings of ACL*, 2016.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, et al. On the properties of neural machine translation: Encoder-decoder approaches. *Proceedings of SSST*, 2014.

[Gerber and Chai, 2010] Matthew Gerber and Joyce Chai. Beyond NomBank: A study of implicit arguments for nominal predicates. In *Proceedings of ACL*, 2010.

[Graves *et al.*, 2014] Alex Graves, Greg Wayne, et al. Neural turing machines. *CoRR*, 2014.

[Graves *et al.*, 2016] Alex Graves, Greg Wayne, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 1997.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of ICLR*, 2014.

[Lin *et al.*, 2016] Yankai Lin, Shiqi Shen, et al. Neural relation extraction with selective attention over instances. In *Proceedings of ACL*, 2016.

[Liu *et al.*, 2013] Chunyang Liu, Wenbo Sun, et al. Convolution neural network for relation extraction. In *Proceedings of ADMA*, 2013.

[Liu *et al.*, 2015] Yang Liu, Furu Wei, et al. A dependency-based neural network for relation classification. In *Proceedings of ACL-IJCNLP*, 2015.

[Madotto *et al.*, 2018] Andrea Madotto, Chien-Sheng Wu, et al. Mem2Seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of ACL*, 2018.

[Peng *et al.*, 2017] Nanyun Peng, Hoifung Poon, et al. Cross-sentence n-ary relation extraction with graph LSTMs. *TACL*, 2017.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, et al. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*, 2014.

[Qin *et al.*, 2018] Pengda Qin, Weiran Xu, et al. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of ACL*, 2018.

[Quirk and Poon, 2017] Chris Quirk and Hoifung Poon. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of EACL*, 2017.

[Santoro *et al.*, 2018] Adam Santoro, Ryan Faulkner, et al. Relational recurrent neural networks. In *Proceedings of NeurIPS*, 2018.

[Santos *et al.*, 2015] Cicero Nogueira dos Santos, Bing Xiang, et al. Classifying relations by ranking with convolutional neural networks. In *Proceedings of ACL-IJCNLP*, 2015.

[Socher *et al.*, 2012] Richard Socher, Brody Huval, et al. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP*, 2012.

[Song *et al.*, 2018] Linfeng Song, Yue Zhang, et al. N-ary relation extraction using graph-state lstm. In *Proceedings of EMNLP*, 2018.

[Sorokin and Gurevych, 2017] Daniil Sorokin and Iryna Gurevych. Context-aware representations for knowledge base relation extraction. In *Proceedings of EMNLP*, 2017.

[Sukhbaatar *et al.*, 2015] Sainbayar Sukhbaatar, Arthur Szlam, et al. End-to-end memory networks. In *Proceedings of NIPS*, 2015.

[Swampillai and Stevenson, 2011] Kumutha Swampillai and Mark Stevenson. Extracting relations within and across sentences. In *Proceedings of RANLP*, 2011.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, et al. Attention is all you need. In *Proceedings of NIPS*, 2017.

[Vu *et al.*, 2016] Ngoc Thang Vu, Heike Adel, et al. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of NAACL*, 2016.

[Weston *et al.*, 2014] Jason Weston, Sumit Chopra, et al. Memory networks. *CoRR*, 2014.

[Wick *et al.*, 2006] Michael Wick, Aron Culotta, et al. Learning field compatibilities to extract database records from unstructured text. In *Proceedings of EMNLP*, 2006.

[Wu *et al.*, 2017] Yi Wu, David Bamman, et al. Adversarial training for relation extraction. In *Proceedings of EMNLP*, 2017.

[Xiao and Liu, 2016] Minguang Xiao and Cong Liu. Semantic relation classification via hierarchical recurrent neural network with attention. In *Proceedings of COLING*, 2016.

[Xu *et al.*, 2015] Yan Xu, Lili Mou, et al. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of EMNLP*, 2015.

[Yoshikawa *et al.*, 2011] Katsumasa Yoshikawa, Sebastian Riedel, et al. Coreference based event-argument relation extraction on biomedical text. *J. Biomed. Semant.*, 2011.

[Zeng *et al.*, 2014] Daojian Zeng, Kang Liu, et al. Relation classification via convolutional deep neural network. In *Proceedings of COLING*, 2014.

[Zeng *et al.*, 2015] Daojian Zeng, Kang Liu, et al. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of EMNLP*, 2015.

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

[Zhang *et al.*, 2015] Shu Zhang, Dequan Zheng, et al. Bidirectional long short-term memory networks for relation classification. In *Proceedings of PACLIC*, 2015.

[Zhou *et al.*, 2016] Peng Zhou, Wei Shi, et al. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of ACL*, 2016.