Relation Extraction with Temporal Reasoning Based on Memory Augmented Distant Supervision

Jianhao Yan[1], **Lin He**[1], **Ruqin Huang**[2], **Jian Li**[3]*, **and Ying Liu**[1]
[1]Institute for Network Sciences and Cyberspace, Tsinghua University
[2]Department of Computer Science and Technology, Tsinghua University
[3]Institute for Interdisciplinary Information Sciences, Tsinghua University
{yjh16, he-l14, hrq16}@mails.tsinghua.edu.cn
lijian83@tsinghua.edu.cn, liuying@cernet.edu.cn

Abstract

Distant supervision (DS) is an important paradigm for automatically extracting relations. It utilizes existing knowledge base to collect examples for the relation we intend to extract, and then uses these examples to automatically generate the training data. However, the examples collected can be very noisy, and pose significant challenge for obtaining high quality labels. Previous work has made remarkable progress in predicting the relation from distant supervision, but typically ignores the temporal relations among those supervising instances. This paper formulates the problem of relation extraction with temporal reasoning and proposes a solution to predict whether two given entities participate in a relation at a given time spot. For this purpose, we construct a dataset called WIKI-TIME[1] which additionally includes the valid period of a certain relation of two entities in the knowledge base. We propose a novel neural model to incorporate both the temporal information encoding and sequential reasoning. The experimental results show that, compared with the best of existing models, our model achieves better performance in both WIKI-TIME dataset and the well-studied NYT-10 dataset.

1 Introduction

As an important technique to automatically complete the knowledge base and reduce labeling efforts, distant supervision (DS) for relation extraction has drawn much attention. In DS, we align the entity pair $(head, tail)$ from a triple $\langle head, rel, tail\rangle$ extracted from a huge knowledge base (e.g., Freebase, Wikidata) with sentences from free texts (e.g., Wikipedia, New York Times) to obtain the training examples, and the label of such an example is the corresponding relation *rel*. Therefore, DS can automatically create a set of training data for each entity pair.

However, the noisy training data problem (Riedel et al., 2010) significantly affects the performance of DS. Therefore, most of the recent approaches (Riedel et al., 2010; Hoffmann et al., 2011; Zeng et al., 2015; Lin et al., 2016) follow a common assumption called the at-least-once assumption, which treats all aligned sentences of each entity pair as one training sample. We refer to a sentence as an instance and all sentences aligned to one entity pair as a mention set in the following, respectively.

The models in previous work (Zeng et al., 2015; Lin et al., 2016; Luo et al., 2017) generally include two parts, **encoding** and **fusion**. The former encodes each instance into a low-dimensional representation. The latter combines representation of each instance. Then, their combination is used to predict the relation.

Although the approaches mentioned above seem promising, they have the following limitations:

1. They all use a separate but identical encoding module among instances and introduce no difference temporally.
2. They only adopt single step of fusion and introduce no sentence-level reasoning.

We remark that the aforementioned approaches may be enough for the standard NYT-10 dataset (Riedel et al., 2010), because the dataset only extracts instances from New York Times corpus from the year 2005 to 2007 and consists of few mention sets with long time span. However, as one can easily imagine, ignoring temporal information may cause inaccurate predictions, especially when a mention set has a long time span and some

*Jian Li is the corresponding author.
[1]https://github.com/ElliottYan/DS_Temporal

instances express different relations. For example, suppose we want to predict the relation between Angelina Jolie and Brad Pitt (using Wikidata). The knowledge base contains a factual relation of *spouse* between them with the valid period from August 2014 to September 2016. However, the extracted mention set contains instances about their marriage in 2014, as well as their divorce in 2016. Because existing models do not encode temporal information, the relation they extract is likely to be the one with highest confidence. In this example, their models may predict the relation of marriage since the instances may suggest a higher confidence for the relation of marriage. But the correct prediction should be divorce. As shown in the above example, we can see it is necessary to include temporal information in DS.

On the other hand, in fusion module, most existing work focused on denoising using methods such as attention or reinforcement learning. We want to argue that a sentence-level reasoning can also be useful since there are instances which are not direct positive examples for the given relation, but can provide supporting evidence. We call them remote instances. Consider the Jolie-Pitt example again. Suppose we are to predict their relation after their divorce. The instances about their marriage also indirectly help to infer their divorce since marriage is the premise of divorce. Hence, we need an algorithm that can incorporate temporal information and perform reasoning over remote instances.

In this paper, we address both limitations and extend the task to predict the relation of a particular entity pair at any specific time spot. The problem can be formulated as a sequence labeling problem (See § 2). We propose a novel relation extraction architecture that can address both aforementioned limitations. Our model follows the popular encoding-fusion architecture, but makes two crucial modifications. Firstly, we introduce temporal encoding to model the temporal information among the instances in the encoding. Secondly, we use the Memory Network (Sukhbaatar et al., 2015; Miller et al., 2016) to iteratively reason over temporally augmented encodings in the fusion part.

Moreover, we evaluate our model on the widely studied NYT-10 dataset (Riedel et al., 2010) and a new WIKI-TIME dataset. The construction of WIKI-TIME is similar to that of the NYT-10 dataset except for two important differences. One is that we only consider triples $\langle head, rel, tail\rangle$ with the valid period $(T1, T2)$. For example, the triple $\langle Jolie, married, Pitt\rangle$ has a valid period of $(2014.08, 2016.09)$. The other is that we extract contextual temporal information for each aligned instance. We use Wikidata (Vrandečić and Krötzsch, 2014) as knowledge base and Wikipedia as free corpus. Both automatic and manual evaluation are applied in the experiments. The experimental results show that, compared with existing models, our model can achieve comparable/better performance in both WIKI-TIME and standard NYT-10 datasets.

Our main contributions can be summarized as follows:

- We introduce a new task aiming to solve the problem of relation extraction with temporal information.
- We propose a novel relation extraction architecture, which encodes both the temporal and semantic information and includes remote instances for temporal reasoning.
- We construct a new WIKI-TIME dataset by aligning Wikidata to Wikipedia, which is specially designed for the task of relation extraction with temporal information.
- The experiment results show that, compared with the best of existing models, our model achieves comparable/better performance both in WIKI-TIME dataset and standart NYT-10 dataset.

2 Formulation

2.1 Traditional Distant Supervision (DS)

The traditional distant supervision (DS) task can be defined as:

Given two entities $\langle head, tail\rangle$ and their corresponding mention set $S = \{s_1, s_2, \cdots, s_T\}$, where s_i denotes the ith instance, the task aims to predict the probability for specific relation r of $\langle head, tail\rangle$:

$$P(r|S = \{s_1, s_2, \cdots, s_T\}). \tag{1}$$

The task can be seen as a multi-label multi-instance classification problem.

2.2 Distant Supervision with Temporal Reasoning

In distant supervision with temporal reasoning, our goal is to predict the relation between two entities at any specific time spot. Because modeling over any specific time spot is non-trivial, we relax the goal to predict the relation between two given entities at any mentioned time spots. Note that we can infer the relations at other time spots using prediction at mentioned ones. Formally, the relation r_t at $t \in (t_1, t_2]$ can be infered by r_{t_1}. Therefore, we can model the problem as a sequence labeling problem with noisy inputs.

Given two entities, we collect the chronologically sorted list of its mention instances and the time spot associated with each instance . We denote the list by $S = \{(s_1, t_1), \cdots, (s_T, t_T)\}$, where (s_i, t_i) is the ith instance and the associated time spot. Our goal is to predict the probability of relation r at time spot t_i:

$$P(r_{t_i} | S = \{(s_1, t_1), \cdots, (s_T, t_T)\}, t_i). \qquad (2)$$

3 Methodology

3.1 Overview of TempMEM

Note that RNN-like models are not suitable for this sequence labeling problem, because the input sequence contains noisy sentences and lacks direct dependency between time steps. We propose a neural model called TempMEM which models the sequence labeling problem by creating query sequence based on each mentioned time spot.

TempMEM also follows the encoding-fusion framework (Zeng et al., 2015; Lin et al., 2016; Luo et al., 2017). However, we make two crucial modifications to the original framework. First, for the encoding part, we use time-aware encoding modules for instances instead of identical ones. Second, we use the memory network to iteratively reason over instances, which makes use of remote instances.

In the following sections, we introduce how to encode the temporal and semantic information and include remote instances for temporal reasoning.

3.2 Encoding

3.2.1 Sentence Encoding

For sentence encoding, here we apply the Convolutional Neural Network (CNN) and the Piecewise Convolutional Neural Network (PCNN) (Zeng et al., 2015). Note that, since TempMEM has no preferance over specific sentence encoding, other encoding modules like word memory (Feng et al., 2017) or self-attention can also be used here.

The inputs of convolution layers are word embeddings concatenated with position features. For a detailed description of the inputs, we refer the readers to (Zeng et al., 2015).

First, the convolution layer extracts local features with sliding window $\hat{w}$ over the input representation. Formally, the convolution operates on the concatenation of the input representations $\boldsymbol{X}_{k:k+\hat{w}}$ of instance j with the shared parameters $\boldsymbol{W}_c \in \mathbb{R}^{D*\hat{w}}$ and $b_c \in \mathbb{R}^1$:

$$o_{c,k} = \boldsymbol{W}_c \cdot \boldsymbol{X}_{k:k+\hat{w}} + b_c, \qquad (3)$$

where $o_{c,k}$ is the kth output of channel c.

Then, we use the piece-wise max-pooling layer. It divides the outputs of filters into three parts $\{\boldsymbol{o}_{c,0:h}, \boldsymbol{o}_{c,h:t}, \boldsymbol{o}_{c,t:N}\}$ and performs max-pooling over each part:

$$\boldsymbol{o}_c = [\max_{0 \le k < h}(o_{c,k}), \max_{h \le k < t}(o_{c,k}), \max_{t \le k < N}(o_{c,k})], \qquad (4)$$

where h and t denote the indices of the head and tail entities, respectively. The concatenation of the output of all channels c is considered as the convolutional representation of instance j:

$$\boldsymbol{O}_j = [\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_C], \qquad (5)$$

where C denotes the number of filters.

3.2.2 Temporal Encoding

In order to introduce temporal priorities among instances, it is necessary to inject temporal information into the encoding part. We want the temporal encoding to have the following characteristics:

- The temporal encodings should comply with the chronological order of instances.
- The difference between two time spots determines the similarity between two temporal encodings.

Since directly encoding the time spot value leads to huge difference among mention sets of the dataset, we propose an approximate approach with PE encoding (Vaswani et al., 2017) based on the rank (i.e. position of an instance in a mention set with chronological order):

$$\boldsymbol{PE(j)} = \begin{cases} sin(j/10000^{d/d_m}) & if\ d\%2 = 0 \\ cos(j/10000^{(d-1)/d_m}) & if\ d\%2 = 1 \end{cases}, \qquad (6)$$

where j is the rank of instance s, d is the dimension, and d_m is the dimension of temporal encoding. Obviously, the PE encoding complies with the chronological order and the similarity between

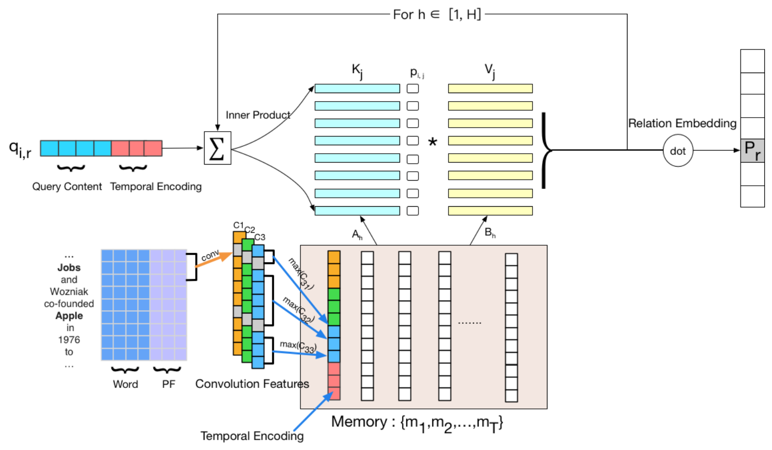


Figure 1: Overall TempMEM architecture

two PE encodings (by dot product) is determined by their rank difference.

Then, we concatenate the corresponding temporal encoding with the convolutional features of instance j to form the final representation of each instance with a learnable scale factor λ:

$$\boldsymbol{m}_j = [\boldsymbol{O}_j; \lambda \cdot \boldsymbol{PE(j)}]. \tag{7}$$

3.3 Fusion

In the fusion part, we use the Memory Network to perform temporal reasoning among different instances. Each encoded instance is considered to be a memory slot. Then, we construct a time specific query and iteratively compute the weighted attention over all instances. We detail the process in the following sections.

3.3.1 Query Construction

We construct each query with the guidance of the following intuition.

- Relation extraction within instances is equal to the query "what is the *relation* between *head* and *tail* at time spot t_i?".

So, we construct our queries based on four key variables, $(relation, head, tail, t_i)$.

Specifically, we combine the embeddings (pre-trained by TransE (Bordes et al., 2013)) of *head* and *tail* and project the combination through an affine matrix $\boldsymbol{\Phi}_q \in \mathbb{R}^{D_e * D_r}$, where D_e and D_r denote the dimension of relation and entity embedding, respectively. After the projection, we add the randomly initialized relation embedding. The formal definition of a query is given below:

$$\boldsymbol{q}_r = \boldsymbol{R}_r + (\boldsymbol{E}_{head} + \boldsymbol{E}_{tail}) * \boldsymbol{\Phi}_q, \tag{8}$$

where $\boldsymbol{R}_r \in \mathbb{R}^{D_r}$ is the embedding of specific relation r and $\boldsymbol{E}_* \in \mathbb{R}^{D_e}$ is the entity embedding. Finally, we also concatenate the query with the same temporal encoding defined in § 3.2.2 to obtain the ith query:

$$\boldsymbol{q}_{r,i} = [\boldsymbol{q}_r; \lambda \cdot \boldsymbol{PE(i)}]. \tag{9}$$

3.3.2 Iterative Reasoning

In this part, we introduce how to use the queries to perform temporal reasoning. Two operations are involved, memory addressing and reading.

One of our key motivations is to consider the remote instances. So, instead of using single step attention computation as in previous work (Lin et al., 2016; Luo et al., 2017; Ji et al., 2017), we perform an overall H steps of memory addressing and reading to obtain the final prediction. Within each step (also called hop), we update the query value by adding the output of the previous step, which provides a gradual shift in attention. Next, we introduce the whole process in detail.

Memory Addressing In addressing, we compute the similarity between the query vector $\boldsymbol{q}_{i,r}$ and each candidate memory slot key $\boldsymbol{K}_j$. Note that the encoding output $\boldsymbol{m}_j$ is not in the same continuous space as the query vector. So, we adopt linear projections to both memory keys:

$$\boldsymbol{K}_j = \boldsymbol{A}_h^T \cdot \boldsymbol{m}_j, \tag{10}$$

where $\boldsymbol{A}_h \in \mathbb{R}^{D_m * D_r}$. Then, we compute the similarity score and importance probability using the bilinear form,

$$s_{i,j} = \boldsymbol{q}_{i,r}^T \cdot \boldsymbol{W}_a \cdot \boldsymbol{K}_j, \tag{11}$$

$$p_{i,j} = \frac{exp(s_{i,j})}{\sum_{\hat{j}=1}^{M} exp(s_{i,\hat{j}})}, \tag{12}$$

where $\boldsymbol{W}_a \in \mathbb{R}^{D_r * D_m}$ is the model parameter to be learned and i, j are the indices of queries and memory slots.

As for the addressing step, it worths noting that the query and memory slots are both concatenated with temporal encodings. If we define the embedding layer $\boldsymbol{A}$ as the identity matrix, each similarity score of a query-memory pair can be divided into two parts,

$$s_{i,j} = \boldsymbol{q}_i^T \cdot \boldsymbol{o}_j + \lambda^2 \cdot \boldsymbol{PE(i)}^T \cdot \boldsymbol{PE(j)}. \quad (13)$$

Each query can automatically attend to instances with either close encoding representations or close temporal encodings. This tradeoff also accords with our intuition, since the confidence of a relational factual statement decreases when the time span increases.

Memory reading The value of each memory slot, which is also projected by an affine matrix $\boldsymbol{B} \in \mathbb{R}^{D_m * D_r}$, is read by computing the weighted sum over all memory slots with the importance probability derived in the addressing step:

$$\hat{\boldsymbol{q}}_i = \sum_j p_{i,j} \boldsymbol{V}_j, \quad (14)$$

where $\boldsymbol{V}_j = \boldsymbol{B}_h^T \cdot \boldsymbol{m}_j$.

Iterative computation Here, we combine the above two operations as a single step for reasoning. We use $h \in [1, H]$ to denote a particular step, where H is the total step number. To achieve a step-by-step reasoning, we update the next step query q^{h+1} with the summation of the current step output $\hat{q}^h$ and the current query q^h:

$$\boldsymbol{q}^{h+1} = \boldsymbol{q}^h + \hat{\boldsymbol{q}}^h. \quad (15)$$

During training, we add dropout with probability p_b at the final query step. By combining the previous hop query and the output in this way, TempMEM can gain information from the last read output and shift addressing attention to remote instances.

3.3.3 Output

In the output module, we define the conditional probability $P(r|S, \theta)$ through a softmax layer as follows:

$$P(r|S, \theta) = \frac{exp(\boldsymbol{R}_r^T \cdot \boldsymbol{q}_r^H)}{\sum_{\hat{r}=1}^{N_r} exp(\boldsymbol{R}_{\hat{r}}^T \cdot \boldsymbol{q}_r^H)}, \quad (16)$$

where N_r is the total amount of pre-defined relations. Also, since we construct and predict relations using the same relation embedding in the query construction part, we remove the original query from the last hop query q^H to keep the prediction unbiased.

3.4 Optimization

Here we introduce the learning and optimization details of TempMEM. We use query-level CrossEntropy loss as our objective function:

$$J(\theta) = \sum_{s=1}^{N_s} \sum_{i=1}^{T} y_t \cdot \log p(\hat{y}_t | S_s, \theta, t_i), \quad (17)$$

where N_s is the number of sets and T is the length of query sequence.

We use stochastic gradient descent (SGD) to minimize our objective function. For the exploration of optimization, we add small white noise to the gradients (Neelakantan et al., 2015). We also anneal the learning rate l by $0 < \rho < 1$ (i.e., $l \leftarrow \rho \cdot l$) for every τ epochs.

4 Experiment

4.1 Dataset

We evaluate our model on two datasets, the widely used NYT-10 dataset which is developed by (Riedel et al., 2010) and the WIKI-TIME dataset we created.

4.1.1 NYT-10

This dataset is generated by aligning Freebase entities to New York Times corpus (NYT) of years from 2005 to 2007. There are 53 pre-defined relations including a particular relation NA which indicates no relation between *head* and *tail*. The training data contains 522,611 sentences, 281,270 entity pairs, and 18,252 of them are relational facts. The testing data contains 172,448 sentences, 96,678 entity pairs, and 1,950 of them are relational facts.

4.1.2 WIKI-TIME

Similar to NYT-10, the WIKI-TIME dataset is also generated by aligning knowledge base entities to free corpus, except that we choose Wikidata and Wikipedia instead of Freebase and NYT news.

The motivation of creating WIKI-TIME is to generate a time aligned dataset that can support temporal reasoning. Hence, we filter knowledge base entities that participate in relations with informative temporal features, such as start time, end time. Besides, we tag the aligned sentences with their time expressions in contexts. Then, we align the contextual time expressions with the valid period of each relation to achieve labeling. For example, sentence like:

"On September 19, 2016, Jolie filed for divorce from Pitt, citing irreconcilable differences."
is labeled with no relation (NA).

The dataset contains 57 relations. The training set contains 97,616 sentences and 20,085 entity pairs. The test set contains 39,990 sentences and 8,641 entity pairs. [2]

4.2 Experiment Details

Hyper-Parameter Settings For WIKI-TIME experiments, we construct query over each appeared time spot in the mention set. On the other hand, for NYT-10 experiments, we adopt a single query without temporal encoding to compare results with other baseline methods since the dataset only contains one label for each mention set.

Among all experiments, we use 230 convolution kernels with windows size 3. The dropout probability p_d is set to 0.5. We try various max hops values H (from 1 to 5) to test how reasoning works in our model. We train the models with 20 epochs and 50 epochs for NYT-10 dataset and WIKI-TIME dataset and report the best performance.

As for optimization step, we adopt SGD with gradient plus Gaussian noise with standard deviation of 0.01, which helps to better generalize. Also, we apply gradient decay of rate ($\rho = 0.5$) over every $\tau = 10$ epochs. The learning rates for NYT-10 and WIKI-TIME experiments are set to 0.001 and 0.01, respectively.

4.2.1 Input Vectors

With regard to inputs, we use 50-d Glove (Pennington et al., 2014) word embeddings pretrianed on Wikipedia and Gigaword and 5-d position embedding. The temporal encodings are either initialized with random 50-d vectors, which are learned during training, or set directly with PE. For entity embeddings, we use the TransE (Bordes et al., 2013) entity vectors pretrained on Wikidata released by the OpenKE platform.

4.2.2 Evaluation Metric

The performance of comparative experiment is reported by precision-recall (PR) curve. Specifically, we sort the prediction scores of the model in descending order (without NA relation) and compute the precision with threshold for each recall value. Also, we report the $P@N$ values which indicate the precision over N predictions with the highest confidence scores.

[2] The details of construction of WIKI-TIME can be found in *Appendix A*.

Symbol	Remarks
CNN_ONE	(Zeng et al., 2015).
CNN_ATT	(Lin et al., 2016).
CNN_AVE	(Lin et al., 2016).
TempMEM	Our model without temporal encoding.
TempMEM+R	Our model with random intialized temporal encoding.
TempMEM+P	Our model with PE encoding.

Table 1: Notations.

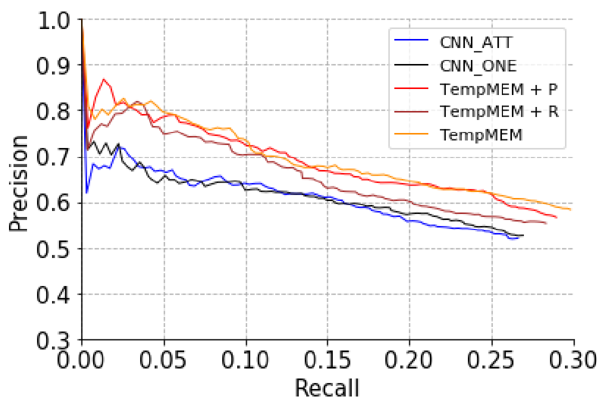


Figure 2: Precison-Recall curve of bag-level experiment on WIKI-TIME. Best viewed in color.

4.3 Performance of iterative reasoning

To test the effect of iterative reasoning over instances, we implement the neural models proposed in previous work (Zeng et al., 2015; Lin et al., 2016), from the source code released by authors. Since the previous models perform prediction in bag-level, the label is given by the latest relation appeared in KB. As for our models, we fix the number of hops $H = 2$ and set the encoding to CNN. [3] The notations of the experiments are shown in Table 1.

As shown in Figure 2 and Table 2, we have the following observations: (1) All TempMEM models achieve better performance compared with the previous neural models (CNN_ONE, CNN_ATT).

[3] The PCNN encoding is not used in WIKI-TIME dataset. The detailed explanation is given in *Appendix A*.

Method	P@N_100	P@N_200	P@N_300
CNN_ATT	67.33	67.66	66.45
CNN_ONE	70.3	68.66	65.78
TempMEM	81.18	**82.09**	**78.41**
TempMEM+R	79.21	78.61	75.42
TempMEM+P	**81.19**	79.1	77.41

Table 2: Comparison with previous models. P@N_100/200/300 refers to the precision for the highest 100, 200 and 300 predictions in WIKI-TIME.

Method	Bag-level F1	Query-level F1
CNN_ATT	39.66	-
CNN_ONE	40.15	-
TempMEM	47.88	54.75
TempMEM+R	46.76	47.83
TempMEM+P	**54.86**	**60.01**

Table 3: Manual evaluation of Bag-level and Query-level F1 scores in WIKI-TIME.

Recall that the hop number is set to 2. This can be seen as an ablation experiment. The results suggest that the remote instances can generally help relation extraction task. (2) TempMEM + P clearly outperforms TempMEM + R, which proves that the properly chosen temporal encodings help the performance.

Note that, in the columns "P@N_200" and "P@N_300" of Table 2, we find that the pure TempMEM outperforms TempMEM + P and TempMEM + R. Based on the results in Table 3, their drop of performance comes from the noisy labeling problem of distant supervision.

Also, TempMEM can catch relation changes through the timeline of two entities. We refer the readers to the case study in *Appendix B*.

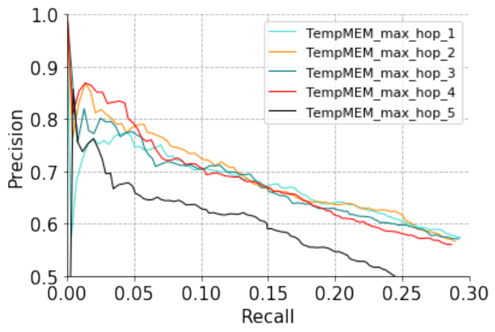


Figure 3: Number-of-hops experiment on WIKI-TIME. Best viewed in color.

4.4 Manual Evaluation on WIKI-TIME

Since the WIKI-TIME is distantly collected, we want to obtain a more precise view of how the models perform. So, we apply the manual evaluation to verify our experimental results. We randomly pick 200 mention sets in the test set of WIKI-TIME and ask two annotators to label the relation for each instance. The annotation rule is to label the instance with the relation that can be inferred from the instance itself or previous instances. As shown in Table 3, the manual evaluated F1 scores are basically consistent with the PR curves in Figure 2, which indirectly proves the WIKI-TIME's quality. Also, we find that the TempMEM + P achieves the best performance and shows obvious advantages in both query-level and bag-level F1 scores over the naive TempMEM (i.e., with no temporal encodings). This proves the effectiveness of our temporal encodings.

4.5 Effect of the Number of Hops

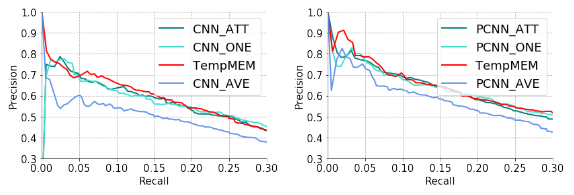
In this section, we discuss the effect of different number of hops in TempMEM. We change the hop value from 1 to 5 and evaluate the precision and recall of our models in query-level. The hyper-parameters are fixed. The temporal encoding is set to PE and each model is trained for 30 epochs.

The results of the hop number experiment are depicted in Figure 3. From the results, we can observe that models show better performance with hop number 2 and 4. Most of the improvement of the model with hop number 4 resides in the recall range [0, 0.05], but the performance remains in the similar trend with other models in the recall range [0.05, 0.2]. In addition, we notice that the performance of the models fluctuates with the increase in the number of hops and the model with even hop number generally perform better than its predecessor, e.g. models with hop number 4 and 5. We believe that the reason might lie in the distribution of the hop distance between origin instance and useful remote instance.

4.6 Performance on NYT-10 Dataset

In this section, we report our results on the well-studied NYT-10 dataset. By evaluating our model in the NYT-10 dataset, our objective is to prove the power of reasoning among remote instances.

Note that, in the NYT-10 dataset, there is no temporal information for each instance, so we only use one query for each mention set and there's no



(a) Results with CNN encoding. (b) Results with PCNN encoding.

Figure 4: Precison-Recall curve on **NYT-10**. Best viewed in color.

temporal encoding for each instance. Also, we do not use the entity embedding for the NYT-10 experiments.

The results are shown in Figure 4. For both CNN and PCNN models, We can see that our models exceed the performance of all other models (CNN_ATT, CNN_ONE, CNN_AVE, PCNN_ATT, PCNN_ONE, PCNN_AVE) in the range of low recall values. In the high recall range, our models also have results about the same as the best model among others. This suggests that even without the temporal encoding, reasoning over remote instances is indeed useful in relation extraction task.

5 Related Work

5.1 Distant Supervision

Distant supervision for relation extraction is an important, automatic method of completing knowledge base.

(Riedel et al., 2010) made the at-least-once assumption that led the distant supervision for relation extraction to multi-instance learning. (Hoffmann et al., 2011) and (Surdeanu et al., 2012) tried to model the task with a multi-instance, multi-label setting using the classical graph model.

Recently, some work focused on applying deep neural network to the DS task. (Zeng et al., 2014) was the first trial to apply deep learning in relation extraction by solving a classification problem with fully supervised approach. (Zeng et al., 2015) moved a step further and introduced the multi-instance learning paradigm by using only the most important instance to predict relation. (Lin et al., 2016; Liu et al., 2017; Ji et al., 2017) improved the previous work by adding attention mechanism to instances and automatically reducing the weights of noisy instances. There are other approaches that tried to reduce the impact of noise in DS by using active learning (Sterckx et al., 2014) and reinforcement learning (Feng et al., 2018).

However, previous work focused on denoising but ignored the exploration of the remote instances and introduced no temporal information to support relation extraction. In this paper, we introduce temporal information into DS and combine it with the memory network to perform reasoning over instances.

(Feng et al., 2017) also used the memory network in the context of distant supervision. Their work performed word-level and relation-level reasoning to model the importance of words and dependency between relations. Their motivation was to gain better sentence encoding and relation modeling, while in our model, we apply the sentence-level memory network to understand the inference process among instances.

5.2 Temporal Relation Extraction

Also, this work is related to temporal relation extraction. (Dligach et al., 2017) was the first approach to use neural models for temporal relation extraction. (Tourille et al., 2017) used a Bi-LSTM to identify narrative containers between events and time expressions. (Cheng and Miyao, 2017) introduced dependency paths and used a ”common-root” to solve the cross-sentence dependency. (Meng and Rumshisky, 2018) leveraged the Neural Turing Machine to enhance context-awareness of the temporal relation extraction model.

Previous work in temporal relation extraction was dedicated to event timelining and focused on dealing with relations between event and time expression. In constrast, our model aims to solve general entity to entity relation extraction by instance-level temporal reasoning based on a coarse-grained timelining.

5.3 Temporal Slot Filling

Another related research aspect is the temporal slot filling (TSF) task introduced in knowledge base population (Surdeanu, 2013; Ji et al., 2014). Distant supervision approaches (Garrido et al., 2013; Cucerzan and Sil, 2013; Sil and Cucerzan, 2014) (from Freebase and Wikipedia infoboxes) are widely applied to address the lack of supervising data. (Reinanda and De Rijke, 2014) performed the prior-sampling on distant supervision data to correct the mismatch of distributions.

The TSF task is similar to the task defined in this paper in the sense that TSF also asks the model to identify the start and end date of one knowledge triple $\langle head, rel, tail \rangle$. The difference is that the TSF task's objective is to predict validity period given the *head*, *rel* and *tail*, while in our setting, we predict *rel* between two entities in different periods.

6 Conclusion

In this paper, we formulate the task of distant supervision with temporal relation reasoning by modeling it as a sequence labeling problem. Following the DS paradigm, we created a new dataset called WIKI-TIME which is designed for the temporal relation extraction task. In addition, we propose an encoding-fusion model, TempMEM, which combines both encoding and reasoning temporally. At each computation step, our model can automatically attend information with either close representation or close temporal encoding. In experiments, we compare our model with the existing methods in both the well-known NYT-10 dataset and our WIKI-TIME dataset. Both automatic and manual evaluation are applied in the experiments. The experimental results show that our model not only realizes better performance in relation extraction by introducing instance-level reasoning but also improves the reasoning by bringing the temporal information in.

In the future, we plan to further explore the effect of different encoding modules like Bi-LSTM or self-attention and try to model temporal information with more sophisticated choices.

Acknowledgments

The research is supported in part by the National Basic Research Program of China Grant 2015CB358700, the National Natural Science Foundation of China Grant 61822203, 61772297, 61632016, 61761146003, and a grant from Microsoft Research Asia.
Also, we want to thank the very kind comments from anonymous reviewers, which help improve this paper.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Fei Cheng and Yusuke Miyao. 2017. Classifying temporal relations by bidirectional lstm over dependency paths. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 1–6.

Silviu Cucerzan and Avirup Sil. 2013. The msr systems for entity linking and temporal slot filling at tac 2013. In *TAC*.

Dmitriy Dligach, Timothy Miller, Chen Lin, Steven Bethard, and Guergana Savova. 2017. Neural temporal relation extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 746–751.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data.

Xiaocheng Feng, Jiang Guo, Bing Qin, Ting Liu, and Yongjie Liu. 2017. Effective deep memory networks for distant supervised relation extraction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*, pages 19–25.

Guillermo Garrido, Anselmo Penas, and Bernardo Cabaleiro. 2013. Uned slot filling and temporal slot filling systems at tac kbp 2013: System description. In *TAC*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Guoliang Ji, Kang Liu, Shizhu He, Jun Zhao, et al. 2017. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *AAAI*, pages 3060–3066.

Heng Ji, Taylor Cassidy, Qi Li, and Suzanne Tamang. 2014. Tackling representation, annotation and classification challenges for temporal knowledge base population. *Knowledge and Information Systems*, 41(3):611–646.

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.

Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795.

Bingfeng Luo, Yansong Feng, Zheng Wang, Zhanxing Zhu, Songfang Huang, Rui Yan, and Dongyan Zhao. 2017. Learning with noise: enhance distantly supervised relation extraction with dynamic transition matrix. *arXiv preprint arXiv:1705.03995*.

Yuanliang Meng and Anna Rumshisky. 2018. Context-aware neural model for temporal information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Arvind Neelakantan, Luke Vilnis, Quoc V Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach, and James Martens. 2015. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*.

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

Ridho Reinanda and Maarten De Rijke. 2014. Prior-informed distant supervision for temporal evidence classification. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 996–1006.

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.

Avirup Sil and Silviu-Petru Cucerzan. 2014. Towards temporal scoping of relational facts based on wikipedia data. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 109–118.

Lucas Sterckx, Thomas Demeester, Johannes Deleu, and Chris Develder. 2014. Using active learning and semantic clustering for noise reduction in distant supervision. In *4e Workshop on Automated Base Construction at NIPS2014 (AKBC-2014)*, pages 1–6.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Mihai Surdeanu. 2013. Overview of the tac2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *TAC*.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 455–465. Association for Computational Linguistics.

Julien Tourille, Olivier Ferret, Aurelie Neveol, and Xavier Tannier. 2017. Neural architecture for temporal relation extraction: A bi-lstm approach for detecting narrative containers. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 224–230.

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 6000–6010.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

A Appendix A : WIKI-TIME Construction

Here we illustrate the construction process of WIKI-TIME dataset in detail. The construction of WIKI-TIME follows common distant supervision framework. In distant supervision, we align the knowledge base to free corpus. The chosen knowledge base and free corpus are Wikidata and Wikipedia, respectively.

As depicted in Figure 5, the construction of WIKI-TIME consists of the following procedures.

1. We extract the relations with the valid period (i.e., $\langle t_s, t_e \rangle$) that appeared in WikiData.
2. Based on the extracted relation set, we further extract entity pairs that participate in any relation in the set. Here, each extracted triple should have a valid period (i.e., $\langle E_1, rel, E_2 \rangle$: $\langle t_s, t_e \rangle$).
3. We tag each sentence in Wikipedia corpus using either time expression appeared in

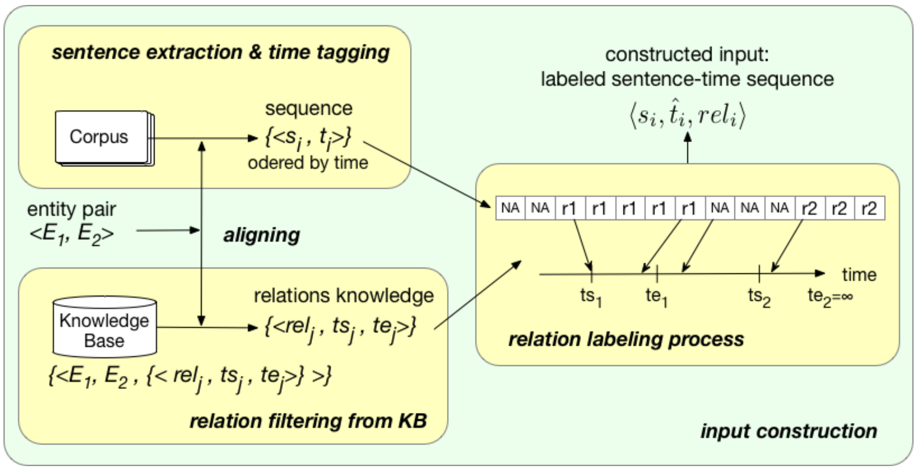


Figure 5: Preprocess of inputs

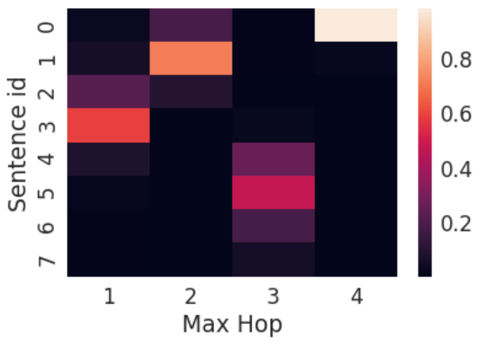
the sentence or last appeared time expression. We implement our own time expression recognition on the basis of NLTK-contrib's timex tool [4] with some modification according to our needs. Each tagged sentence is denoted by tuple $\langle s_i, \hat{t}_i \rangle$.

4. We align the extracted entity pairs to tagged corpus to retrieve our final mention set. Note that we additionally take sentences mentioning entity E_2 in the wiki page of entity E_1 as training sentences for (E_1, E_2) to obtain a larger dataset. In this case, the position of E_1 may not exist in our dataset. Because PCNN splits sentences by positions of entities, it is unclear how to apply it directly to our WIKI-TIME dataset.
5. We order the extracted sentences $\{\langle s_1, \hat{t}_1 \rangle, \cdots \langle s_m, \hat{t}_m \rangle\}$ into a timeline and align them with the corresponding knowledge triple for sentence-level relation $\langle s_i, \hat{t}_i, rel_i \rangle$.

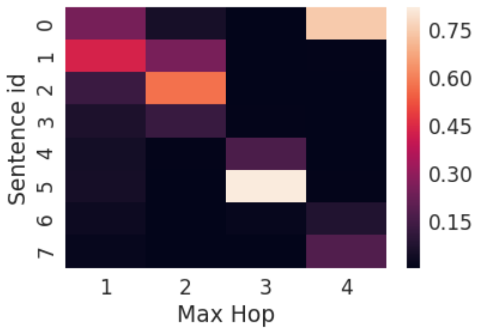
The final dataset contains 57 relations. The training set contains 97,616 sentences and 20,085 entity pairs. The test set contains 39,929 sentences and 8,641 entity pairs. The train/test split is done on entity pair level, so there is no overlap between train and test set.

B Appendix B : Case Study

Table 4 shows an example of our tagged data. For each sentence, we show the corresponding time slot and its relation. Also, the mentions of the entity are highlighted with the bold font and each sentence is indexed by *id* for clearness.



(a) q_3 over relation *Spouse*



(b) q_0 over relation *Spouse*

Figure 6: Attention weights for queries with $H = 4$

For each query, we show its corresponding heat map of attention values over different hops. The case we choose is $\langle Stelios\ \ Kazantzidis, spouse, Marinella \rangle$. In Figure 6 (a), the attention of q_3 focuses on sentences with *id* 2 and 3 which support the relation *spouse* a lot. Then in the next hop, its attention shifts to sentences with *id* 0 and 1, which are the "remote" instances. Then in the third hop, its attention shifts again toward another part

[4] https://github.com/nltk/nltk_contrib/blob/master/nltk_contrib/timex.py

id	Relation	Time Spot	Sentence
0	NA	1957-01-01	Her early career was marked by her collaboration with singer **Stelios Kazantzidis**.
1	NA	1960-01-01	... instances of **Marinella** in films of Greek cinema, from the 1960 by 1966 with **Stelios Kazantzidis** ...
2	Spouse	1964-05-07	**Marinella** married **Stelios Kazantzidis** on 7 May 1964 ...
3	Spouse	1964-05-07	**Stelios Kazantzidis** married **Marinella** on 7 May 1964 ...
4	NA	1966-09-01	In September 1966 he divorced **Marinella**...
5	NA	1968-01-01	Following **Marinella**'s departure Litsa Diamandi ...
6	NA	1968-01-01	**Marinella** sang on some songs ...
7	NA	1968-01-01	**Marinella** had an "answer back" to that latter song ...

Table 4: Aligned sentences of ⟨ Stelios Kazantzidis , Marinella ⟩

of the sentence set (*id* 4 to 6), which contains sentences related to divorce, departure and so on. These findings prove the claim that our model can achieve more accurate relation extraction by exploitation of remote instances.

Also, we demonstrate that the model has different attentions for queries at different time spots. In Figure 6 (b), we still observe the case of ⟨ *Stelios Kazantzidis, spouse, Marinella* ⟩. But, the attention of query q_0 focuses on a different path compared with query q_3. It first focuses on the remote instances with *id* 0 and 1. Then the model shifts to the supporting sentences. After that, it highly focuses on sentence 5, which supports two entities' departure. All these findings show that the temporal encodings significantly affect the result of attention.

The differences between the attentions of quries q_0 and q_3 show that the temporal encodings significantly affect the result of attention.