Refining Sample Embeddings with Relation Prototypes to Enhance Continual Relation Extraction

Li Cui[1], **Deqing Yang**[1*], **Jiaxin Yu**[1], **Chengwei Hu**[1],
Jiayang Cheng[1], **Jingjie Yi**[1] **and Yanghua Xiao**[2,3*]
[1]School of Data Science, Fudan University
[2]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
[3]Fudan-Aishu Cognitive Intelligence Joint Research Center, Shanghai, China
fd2014cl@gmail.com
{yangdeqing, jiaxinyu20, cwhu20, chengjy17, jjyi20, shawyh}@fudan.edu.cn

Abstract

Continual learning has gained increasing attention in recent years, thanks to its biological interpretation and efficiency in many real-world applications. As a typical task of continual learning, continual relation extraction (CRE) aims to extract relations between entities from texts, where the samples of different relations are delivered into the model continuously. Some previous works have proved that storing typical samples of old relations in memory can help the model keep a stable understanding of old relations and avoid forgetting them. However, most methods heavily depend on the memory size in that they simply replay these memorized samples in subsequent tasks. To fully utilize memorized samples, in this paper, we employ *relation prototype* to extract useful information of each relation. Specifically, the prototype embedding for a specific relation is computed based on memorized samples of this relation, which is collected by K-means algorithm. The prototypes of all observed relations at current learning stage are used to re-initialize a memory network to refine subsequent sample embeddings, which ensures the model's stable understanding on all observed relations when learning a new task. Compared with previous CRE models, our model utilizes the memory information sufficiently and efficiently, resulting in enhanced CRE performance. Our experiments show that the proposed model outperforms the state-of-the-art CRE models and has great advantage in avoiding catastrophic forgetting. The code and datasets have been released on https://github.com/fd2014cl/RP-CRE.

1 Introduction

As one of the most important tasks in information extraction (IE), relation extraction (RE) has been widely applied in many downstream tasks, such as knowledge base construction and completion (Riedel et al., 2013). The goal of RE is to recognize a relation predefined in knowledge graphs (KGs) for an entity pair in texts. For example, given the entity pair [*Christopher Nolan, Interstellar*] in the sentence *"Interstellar is an epic science fiction film directed by Christopher Nolan"*, the relation *the-director-of* should be recognized by an RE model.

Conventional RE models (Zeng et al., 2014; Zhou et al., 2016; Zhang et al., 2018a) always assume a fixed pre-defined set of relations and perform once-and-for-all training on a fixed dataset. Therefore, these models can not well handle the learning of new relations, which often emerge in many realistic applications given the continuous and iterative nature of our world (Hadsell et al., 2020). To adapt to such a situation, the paradigm of continual relation extraction (CRE) is proposed (Wang et al., 2019; Han et al., 2020; Wu et al., 2021). Compared with conventional RE, CRE focuses more on helping a model keep a stable understanding of old relations while learning emerging relations, which in fact could be precisely modeled by continual learning.

Continual learning (or lifelong learning) systems are defined as adaptive algorithms capable of learning from a continuous stream of information (Parisi et al., 2019), where the information is progressively available over time and the number of learning tasks is not pre-defined. Continual learning remains a long-standing challenge for machine learning and deep learning (Hassabis et al., 2017; Thrun and Mitchell, 1995), as its main obstacle is the tendency of models to forget existing knowledge when learning from new observations (French, 1999), which is called as ***catastrophic forgetting***. Recent works try to address the problem of catastrophic forgetting in three ways, including consolidation-based methods (Kirkpatrick et al., 2017), dynamic archi-

[*]Corresponding author

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 232–243
August 1–6, 2021. ©2021 Association for Computational Linguistics

tecture (Chen et al., 2015; Fernando et al., 2017) and memory-based methods (Lopez-Paz and Ranzato, 2017; Aljundi et al., 2018; Chaudhry et al., 2018), in which memory-based methods have been proven promising in NLP tasks.

In recent years, some memory-based CRE models have made significant progress in overcoming catastrophic forgetting while learning new relations, such as EA-EMR (Wang et al., 2019), MLLRE (Obamuyide and Vlachos, 2019), CML (Wu et al., 2021) and EMAR (Han et al., 2020). Despite of their effectiveness, there are some challenges remaining in current CRE. One noticeable challenge is how to restore the sample embedding space disrupted by the learning of new tasks, given that RE models' performance is very sensitive to the quality of sample embeddings. Another challenge is that most existing CRE models have not fully exploited memorized samples. In order to enhance RE performance and overcome the overfitting problem caused by high replay frequency, the samples memorized in these models usually have the same magnitude as the original training samples (Wu et al., 2021), which is unrealistic in real-world tasks.

Inspired by prototypical networks (Snell et al., 2017) for few-shot classification, we employ ***relation prototypes*** to represent different relations in this paper, which help the model understand different relations well. Furthermore, these prototypes are used to refine sample embeddings in CRE. This process is named as ***prototypical refining*** in this paper. Specifically, the prototype for a specific relation is the average embedding of typical samples labeled with this relation, which are collected by K-means and memorized by our model for future use. The prototypical refining can help our model recover from the disruption of embedding space and avoid catastrophic forgetting during learning new relations, thus enhance our model's CRE performance. Another advantage of prototypical refining is the efficient utilization of memorized samples, resulting in our model's less dependence on memory size.

Our contributions in this paper are summarized as follows:

(1) We propose a novel CRE model which achieves enhanced performance through refining sample embeddings with relation prototypes and is effective in avoiding catastrophic forgetting.

(2) The paradigm we proposed for refining sample embeddings takes full advantage of the typical samples stored in memory, and reduces the model's dependence on memory size (number of memorized samples).

(3) Our extensive experiments upon two RE benchmark datasets justify our model's remarkable superiority over the state-of-the-art CRE models and less dependence on memory size.

2 Related Works

Conventional studies in relation extraction (RE) mainly focus on designing and utilizing various deep neural networks to discover the relations between entities given contexts, including: (1) Convolutional neural networks (CNNs) (Zeng et al., 2014, 2015; Nguyen and Grishman, 2015; Lin et al., 2016; Ji et al., 2017) can effectively extract local textual features. (2) Recurrent neural networks (RNNs) (Zhang and Wang, 2015; Xu et al., 2015; Zhou et al., 2016; Zhang et al., 2018a) are particularly capable of learning long-distance relation patterns. (3) Graph neural networks (GNNs) (Zhang et al., 2018b; Fu et al., 2019; Zhu et al., 2019) build word/entity graphs for cross-sentence reasoning. Recently, pre-trained language models (Devlin et al., 2019) have also been extensively used in RE tasks (Wu and He, 2019; Wei et al., 2020; Baldini Soares et al., 2019), and have achieved state-of-the-arts performance.

However, most of these models can only extract a fixed set of pre-defined relations. Hence, continual relation learning, i.e., CRE, has been proposed to overcome this problem. Existing continual learning methods can be divided into three categories: (1) Regularization methods (Kirkpatrick et al., 2017; Zenke et al., 2017; Liu et al., 2018) alleviate catastrophic forgetting by imposing constraints on updating the neural weights important to previous tasks. (2) Dynamic architecture methods (Chen et al., 2015; Fernando et al., 2017) change architectural properties in response to new information by dynamically accommodating novel neural resources. (3) Memory-based methods (Lopez-Paz and Ranzato, 2017; Aljundi et al., 2018; Chaudhry et al., 2018) remember a few examples in previous tasks and continually replay the memory with emerging new tasks. For CRE, the memory-based methods have been proven most promising (Wang et al., 2019; Han et al., 2020). In addition, in order to accurately represent relations with limited samples, the idea of prototypical networks is introduced into RE(Gao et al., 2019; Ding et al., 2021).

There are also many memory networks proposed to remember information of long periods, such as LSTM (Hochreiter and Schmidhuber, 1997) and memory-augmented neural networks (Graves et al., 2016; Santoro et al., 2016). Besides, a new memory module (Santoro et al., 2018) has demonstrated its success in relational reasoning, which employs multi-head attention to allow memory interaction.

3 Methodology

In this section, we introduce our CRE model in details. At first, we formalize the problem of CRE and the memory module used in our model.

3.1 Task Formalization

In general, a single relation extraction (RE) task is to identify (classify) the relation between two entities expressed in a sentence. Formally, the objective of CRE is to accomplish a sequence of K RE tasks $\{T_1, T_2, \ldots, T_K\}$, where the k-th task T_k has its own training set D_k and relation set R_k. Suppose D_k contains N training samples $\{(x_1, t_1, y_1), \ldots, (x_N, t_N, y_N)\}$ where instance $(x_i, t_i, y_i), 1 \leq i \leq N$ indicates that the relation of entity pair t_i in sentence x_i is $y_i \in R_k$. In fact, each task T_k is an independent multi-classification task to identify various relations in R_k. A CRE model should perform well on extracting the relations in all K tasks after being trained with the samples of these tasks. In other words, the model should be capable of identifying the relation of a given entity pair into $\tilde{R}_k$, where $\tilde{R}_k = \cup_{i=1}^{k} R_i$ is the relation set already observed till the k-th task.

Inspired by current CRE models (Wu and He, 2019; Han et al., 2020), we adopt an episodic memory module to store typical samples of relations that the model has learned in former tasks. The memory module for relation r is represented as a memorized sample set $M_r = \{(x_1, t_1, r), \ldots, (x_O, t_O, r)\}$, where each sample is labeled with r and O is the memory size (sample number). Therefore, the episodic memory for the observed relations in $T_1 \sim T_k$ is $\tilde{M}_k = \cup_{r \in \tilde{R}_k} M_r$.

3.2 Model Learning Pipeline

The learning procedure of our model for a current task T_k is shown in Algorithm 1. The procedure contains four major steps:

Prototype Generation (line 2 $\sim$ 13): We first obtain the prototype $\boldsymbol{p}_r$ of each old relation r in $\tilde{R}_{k-1}$ by averaging the embeddings of memorized samples in M_r with sample encoder $\boldsymbol{E}$ (Section 3.3). These prototypes constitute a prototype set $\boldsymbol{P}_k$, which is used to memorize model's embedding space before training on T_k. Note that the encoder $\boldsymbol{E}$ is continuously changing with tasks, the prototypes of relations need to be regenerated at

Algorithm 1: Training procedure for T_k

```
   Input: $D_k$, $R_k$, $\tilde{R}_{k-1}$, $\tilde{M}_{k-1}$
   Output: $\tilde{R}_k$, $\tilde{M}_k$
1  $\boldsymbol{P}_k \leftarrow \emptyset$;
2  for each $r \in \tilde{R}_{k-1}$ do
3      get $M_r$ from $\tilde{M}_{k-1}$;
4      $\boldsymbol{H}_r \leftarrow \emptyset$;
5      for each $(x_i, t_i, r) \in M_r$ do
6          //get $x_i$'s embedding $\boldsymbol{h}_i$ through $\boldsymbol{E}$;
7          $\boldsymbol{h}_i \leftarrow \boldsymbol{E}(x_i, t_i)$;
8          $\boldsymbol{H}_r \leftarrow \boldsymbol{H}_r \cup \boldsymbol{h}_i$;
9      end
10     //compute $r$'s prototype as the average
         of $\boldsymbol{H}_r$'s embeddings;
11     $\boldsymbol{p}_r \leftarrow Avg(\boldsymbol{H}_r)$;
12     $\boldsymbol{P}_k \leftarrow \boldsymbol{P}_k \cup \boldsymbol{p}_r$;
13 end
14 $\tilde{R}_k \leftarrow \tilde{R}_{k-1} \cup R_k$;
15 $\tilde{M}_k \leftarrow \tilde{M}_{k-1}$;
16 for $i$ = 1 to epochs1 do
17     update $\boldsymbol{E}$ and $\boldsymbol{C}$ according to $\mathcal{L}_1$ on $D_k$;
18 end
19 for each $r \in R_k$ do
20     $\boldsymbol{H}_r \leftarrow \emptyset$;
21     for each $(x_i, t_i, y_i) \in D_k$ do
22         if $y_i = r$ then
23             $\boldsymbol{h}_i \leftarrow \boldsymbol{E}(x_i, t_i)$;
24             $\boldsymbol{H}_r \leftarrow \boldsymbol{H}_r \cup \boldsymbol{h}_i$;
25         end
26     end
27     generate $M_r$ by K-means on $\boldsymbol{H}_r$;
28     $\tilde{M}_k \leftarrow \tilde{M}_k \cup M_r$;
29     $\boldsymbol{p}_r \leftarrow Avg(\boldsymbol{H}_r)$;
30     $\boldsymbol{P}_k \leftarrow \boldsymbol{P}_k \cup \boldsymbol{p}_r$;
31 end
32 feed $\boldsymbol{P}_k$ into $\boldsymbol{M}$;
33 for $i$ = 1 to epochs2 do
34     update $\boldsymbol{E}$, $\boldsymbol{M}$ and $\boldsymbol{C}$ according to $\mathcal{L}_2$ on
         $\tilde{M}_k$ with the prototypical refining
         conducted by $\boldsymbol{M}$;
35 end
```

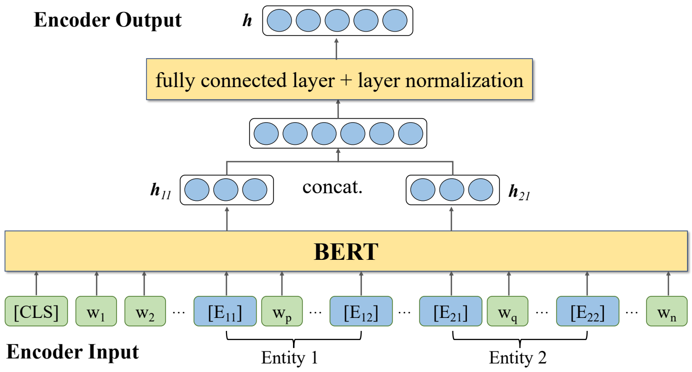


Figure 1: The structure of sample encoder $\boldsymbol{E}$.

the beginning of each task.

Initial Training (line 16 $\sim$ 18): The parameters in sample encoder $\boldsymbol{E}$ and relation classifier $\boldsymbol{C}$ are tuned with the training samples in D_k (Section 3.4).

Sample Selection (line 19 $\sim$ 31): For each relation r in R_k, which is unobserved in the former tasks, we retrieve all samples labeled with r from D_k. Then we use K-means algorithm to cluster these samples. In each cluster, we take the sample closest to the centroid as the memorized typical sample of r, to constitute M_r (Section 3.5). Then, we generate r's prototype $\boldsymbol{p}_r$ based on M_r to expand the prototype set $\boldsymbol{P}_k$.

Prototypical Refining (line 32 $\sim$ 35): To recover the disruption of sample embedding space, which is caused by training on T_k, we use relation prototype set $\boldsymbol{P}_k$ to refine sample embeddings. Specifically, $\boldsymbol{P}_k$ is used to initialize our attention-based memory network $\boldsymbol{M}$ (Section 3.6). The samples in $\tilde{M}_k$ are encoded into embeddings by $\boldsymbol{E}$, and then refined by $\boldsymbol{M}$ before being fed to $\boldsymbol{C}$, to compute the loss function and update model parameters.

In general, the parameter update of our model for T_k includes two stages: (1) Initial training on D_k, where samples are encoded by encoder $\boldsymbol{E}$. (2) Prototypical refining on $\tilde{M}_k$, where sample embeddings are generated by encoder $\boldsymbol{E}$ and then refined by memory network $\boldsymbol{M}$.

Next, we introduce this procedure in detail.

3.3 Sample Encoder

The structure of this sample encoder is displayed in Figure 1, which is used to obtain the embedding of each sample. In our model, the encoder $\boldsymbol{E}$ is built upon BERT (Devlin et al., 2019; Wolf et al., 2020), given its excellent performance on text encoding as a representative pre-trained language model. In addition, entity information has been proven effective in sample encoding for RE tasks (Wu and He, 2019; Baldini Soares et al., 2019). Thus, we highlight the existence of entities in the sentence to augment $\boldsymbol{E}$, through adding special tokens to mark the start and end position of entities. Specifically, we use $[E_{11}]$, $[E_{12}]$, $[E_{21}]$ and $[E_{22}]$ to denote the start and end position of head and tail entity, respectively.

Next, a sample's hidden representation is the concatenation of token embeddings of $[E_{11}]$ and $[E_{21}]$, which has been proven effective in previous works (Baldini Soares et al., 2019). By feeding this concatenation into a fully connected layer along with layer normalization, a sample's final embedding $\boldsymbol{h}$ is generated as follows

$$\boldsymbol{h} = LN\Big(\boldsymbol{W}\big(concat[\boldsymbol{h}_{11}, \boldsymbol{h}_{21}]\big) + \boldsymbol{b}\Big), \quad (1)$$

where $\boldsymbol{h}_{11}, \boldsymbol{h}_{21} \in \mathbb{R}^h$ (h is the dimension of BERT hidden representation) are the hidden representations of $[E_{11}]$ and $[E_{21}]$, $\boldsymbol{W} \in \mathbb{R}^{d \times 2h}$ (d is sample embedding dimension) and $\boldsymbol{b} \in \mathbb{R}^d$ are trainable parameters, and $LN(\cdot)$ is the operation of layer normalization.

3.4 Initial Training for New Task

According to the general assumption of CRE, all relations in R_k are unobserved in former tasks $T_1 \sim T_{k-1}$. We first introduce the model's initial training on a simple multi-classification task.

Specifically, classifier $\boldsymbol{C}$ in our model is a linear softmax classifier. For training set D_k, the loss function is defined as

$$\mathcal{L}_1(\theta) = \sum_{i=1}^{|D_k|} -logP(y_i|x_i, t_i), \quad (2)$$

where $P(y_i|x_i, t_i)$ is calculated by classifier $\boldsymbol{C}$ based on sample (x_i, t_i, y_i)'s embedding output by sample encoder $\boldsymbol{E}$.

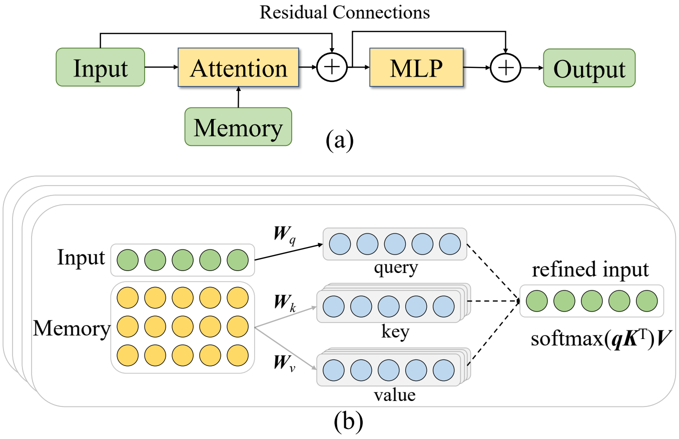


Figure 2: Attention-based memory network $\boldsymbol{M}$. (a) The overall work flow of memory network, where input is the sample embedding generated by $\boldsymbol{E}$. (b) Basic structure of attention heads in attention module.

3.5 Selecting Typical Samples to Memorize Relations

For each relation r in R_k, we select several typical samples into M_r after the initial training with D_K. As the budget of memory is relatively smaller, it is important to select informative and diverse samples to represent r. Inspired by (Han et al., 2020), we apply K-means algorithm upon the embeddings of r's samples, which are generated by sample encoder $\boldsymbol{E}$. Suppose the number of clusters is O, which is also the number of typical samples that we will store to represent r. Then, in each cluster we choose the sample closest to the centroid to represent the cluster and add it into the memory. Such operation ensures that the samples stored in the memory are diverse enough and representative for the relation.

3.6 Refining Sample Embeddings with Relation Prototypes

We propose this module to refine the sample embeddings.

After the initial training for the new task T_k, old relations' embedding space is likely to be disrupted because the model is tuned towards fitting T_k's learning objective (Section 3.4). Instead of just replaying memorized samples for recovery, which is a common practice in continual learning, we refine sample embeddings based on relation prototypes.

Before applying our prototypical refining, we first obtain the prototype embedding $\boldsymbol{p}_r$ for each old relation r in $\tilde{R}_{k-1}$ to constitute the prototype set $\boldsymbol{P}_k$. This step (**Prototype Generation**) is conducted before the initial training for T_k (**Initial Training**) to memorize the former state of our model. Then, we construct an attention-based memory network $\boldsymbol{M}$ based on $\boldsymbol{P}_k$ for prototypical refining, as shown in Figure 2. This network's input is the sample embedding generated by $\boldsymbol{E}$, and its output is fed into $\boldsymbol{C}$ for relation classification. Based on prototypical refining conducted by memory network $\boldsymbol{M}$, our model's embedding space is restored.

Given a sample (x, t, y), its embedding $\boldsymbol{h} \in \mathbb{R}^d$ is generated by $\boldsymbol{E}$ and will be fed to memory network $\boldsymbol{M}$. We also denote the head number of our memory network as N and the hidden dimension of each head as d_1. The output of the i-th attention head is $\boldsymbol{h}_i \in \mathbb{R}^{d_1}$, which is computed as

$$\begin{aligned} \boldsymbol{h}_i &= ATN(\boldsymbol{q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i) \\ &= softmax\left(\frac{\boldsymbol{q}_i \boldsymbol{K}_i^T}{\sqrt{d_1}}\right)\boldsymbol{V}_i, \end{aligned} \tag{3}$$

where $\boldsymbol{q}_i \in \mathbb{R}^{d_1}$ is the linear transformation of input $\boldsymbol{h}$, and $\boldsymbol{K}_i, \boldsymbol{V}_i \in \mathbb{R}^{L \times d_1}$ (L is the current size of $\tilde{R}_k$) is the linear transformation of $\boldsymbol{P}_k$. Then, we concatenate each head's output into the output of multi-head attention layer as

$$\tilde{\boldsymbol{h}} = LN\Big(\boldsymbol{W}_1\big(concat[\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N]\big) + \boldsymbol{h}\Big), \tag{4}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{d \times Nd_1}$ is a trainable matrix.

At last, the final output of $\boldsymbol{M}$ is a residual output computed as

$$\tilde{\boldsymbol{h}}' = LN\Big(\boldsymbol{W}_2\tilde{\boldsymbol{h}} + \tilde{\boldsymbol{h}}\Big), \tag{5}$$

where $\boldsymbol{W}_2 \in \mathbb{R}^{d \times d}$ is also a trainable matrix. $\tilde{\boldsymbol{h}}'$ is the refined embedding of (x, t, y), which incorporating the information of prototypes $\boldsymbol{P}_k$ through Equation 3 and is fed to the classifier $\boldsymbol{C}$.

We take $\tilde{M}_k$ as the training set in this stage and the loss function is

$$\mathcal{L}_2(\theta) = \sum_{i=1}^{|\tilde{M}_k|} -logP(y_i|x_i, t_i), \tag{6}$$

where (x_i, t_i, y_i) is a sample in $\tilde{M}_k$, and $P(y_i|x_i, t_i)$ is calculated by $\boldsymbol{C}$ based on its embedding, which is first generated by $\boldsymbol{E}$ and refined by $\boldsymbol{M}$.

Based on the typicality and diversity of memorized samples (samples that can well represent most samples in this relation), training on $\tilde{M}_k$ can restore the disrupted embedding space of our model with a relatively small computational cost, which allows our model to regain a stable understanding of old relations.

3.7 Prediction

In order to maintain the consistency of training and prediction, our model uses the embeddings refined by $\boldsymbol{M}$ for prediction after training on a new task.

4 Experiments

4.1 Datasets

Our experiments were conducted upon the following two widely used datasets. The training-test-validation split ratio is 3:1:1.

FewRel (Han et al., 2018) It is an RE benchmark dataset originally proposed for few-shot learning, which is annotated by crowd workers and contains 100 relations and 70,000 samples in total. In our experiments, we used the version of 80 relations that has been used (as the training and valid set) for CRE.

TACRED (Zhang et al., 2017) It is a large-scale RE dataset with 42 relations (including *no_relation*) and 106,264 samples built over newswire and web documents. Based on the open relation assumption of CRE, we removed *no_relation* in our experiments. At the same time, in order to limit the sample imbalance of TACRED, we limited the number of training samples of each relation to 320 and the number of test samples of each relation to 40.

4.2 Compared Models

We introduce the following state-of-the-art CRE baselines to be compared with our model in our experiments.

EA-EMR (Wang et al., 2019) maintains a memory to alleviate the problem of catastrophic forgetting.

EMAR (Han et al., 2020) introduces memory activation and reconsolidation for continual relation learning.

CML (Wu et al., 2021) proposes a curriculum-meta learning method to tackle the order-sensitivity and catastrophic forgetting in CRE.

As we adopt pre-trained language model for sample encoding, we replace the encoder (Bi-LSTM) in EMAR with BERT for a fair comparison. This EMAR's variant is denoted as **EMAR+BERT**. Besides, we denote our CRE model with relation prototypes as **RP-CRE** in result display. Since our model only uses the information of memorized samples in attention-based memory network, we further proposed a variant of our model denoted as **RP-CRE+Memory Activation**, by adding a memory activation (Han et al., 2020) step before attention operation, to verify whether more memory replay is needed.

4.3 Experimental Settings

In previous CRE experiments (Wang et al., 2019; Han et al., 2020), relations are first divided into 10 clusters to simulate 10 tasks. However, there are two drawbacks of this setting: (1) Recognizing all relations before training is unrealistic and contrary to the setting of lifelong learning. (2) The relations in one cluster generally have more semantic relevance. Therefore, we adopted a completely random sampling strategy on relation-level in our experiments, which is more diverse and realistic. In addition, the task order of all models is exactly the same.

In the context of continual learning, we pay more attention to the variation trend of models' performance while learning new tasks. Therefore, after training for each new task, we will evaluate the classification accuracy of the models on the test set, which is composed of the test samples of all observed relations.

Given that most recent CRE models are evaluated by distinguishing true relation labels from a small number of sampled negative labels (Wang et al., 2019), which is too simple and rigid for realistic applications. Therefore, we take a rigorous multi-classification task on all observed relations as the evaluation of our model. It is also the reason that the baselines' performance is much

Table 1: Accuracy (%) on all observed relations (which will continue to accumulate over time) at the stage of learning current task, indicating that our model (RP-CRE) significantly surpasses other models and has an advantage in comparison with EMAR+BERT.

FewRel										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR	89.0	69.0	59.1	54.2	47.8	46.1	43.1	40.7	38.6	35.2
EMAR	88.5	73.2	66.6	63.8	55.8	54.3	52.9	50.9	48.8	46.3
CML	91.2	74.8	68.2	58.2	53.7	50.4	47.8	44.4	43.1	39.7
EMAR+BERT	**98.8**	89.1	89.5	85.7	83.6	84.8	79.3	80.0	77.1	73.8
RP-CRE+Memory Activation	98.0	91.4	**91.8**	86.8	87.6	**86.9**	83.7	81.9	80.1	79.5
RP-CRE (Ours)	97.9	**92.7**	91.6	**89.2**	**88.4**	86.8	**85.1**	**84.1**	**82.2**	**81.5**
TACRED										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR	47.5	40.1	38.3	29.9	28.4	27.3	26.9	25.8	22.9	19.8
EMAR	73.6	57.0	48.3	42.3	37.7	34.0	32.6	30.0	27.6	25.1
CML	57.2	51.4	41.3	39.3	35.9	28.9	27.3	26.9	24.8	23.4
EMAR+BERT	96.6	85.7	81.0	78.6	73.9	72.3	71.7	72.2	72.6	71.0
RP-CRE+Memory Activation	97.1	**91.4**	**87.4**	82.1	78.3	**77.8**	74.9	73.5	**73.6**	72.3
RP-CRE (Ours)	**97.6**	90.6	86.1	**82.4**	**79.8**	77.2	**75.1**	**73.7**	72.4	**72.4**

worse than their reported results in the original papers. The method of choosing hyper-parameter for our model is manual tuning. For reproducing our experiment results conveniently, our model's source code, detailed hyper-parameter configurations and processed samples are provided on https://github.com/fd2014cl/RP-CRE.

4.4 Overall Performance Comparison

The performance of our model and baselines are shown in Table 1, where the reported scores are the average of 5 rounds of training. Hyper-parameter configurations of baselines are the same as that reported in original papers. Result of each task is the accuracy on test data of all observed relations.

Based on the results, we find that:

(1) Our strict test and sampling strategy actually increase the difficulties of CRE, causing great difficulties to the compared CRE models. This phenomenon is especially obvious in TACRED that has class-imbalance, even if we have made some restrictions to the number of samples for each relation.

(2) Pre-trained language models, such as BERT, can gain outstanding performance in CRE. Take EMAR for example, replacing Bi-LSTM in it with BERT brings more than 50% of improvement for the last task in FewRel (46.3% to 73.8%), and more than 150% of improvement in TACRED (25.1% to 71.0%). We think this is mainly due to BERT's capability of making rapid migration to new tasks. The remarkable advantage of the BERT-based models in Table 1 in TACRED further justifies BERT's insensitivity to sample imbalance.

(3) Compared with EMAR+BERT, our model also has great advantage, proving that our model can take full advantage of memorized samples and maintain relatively stable performance in continual learning.

(4) Adding memory activation to our models did not significantly improve performance, indicating that it is sufficient to adopt relation prototypes in CRE.

(5) Note that all models have similar performance on the former tasks, but our model obtains more stable performance towards the emergence of new tasks. It implies our model's advantage in long-term memory, which will be proven in Section 4.5.

The average time consumption (on the machine with a single RTX3090) of training RP-CRE is 1h28min, EMAR is 37min and EMAR+BERT is 3h21min. Our model's time consumption is mainly due to the massive parameters of BERT. Given our model's apparent performance improvement with respect to EMAR, such time consumption is relatively acceptable.

Table 2: Accuracy (%) on the test sets from every previous task at the stage of learning the last task (with the same size of memory), indicating that our model has better performance on previous tasks.

Model	T1	T2	T3	T4	T4	T6	T7	T8	T9	T10
RP-CRE (Ours)	**82.8**	**68.4**	**89.0**	**78.8**	**75.7**	**88.1**	**77.3**	**82.9**	**92.3**	90.8
EMAR+BERT	75.2	59.6	77.6	65.8	65.9	80.5	58.9	60.0	87.6	**98.0**

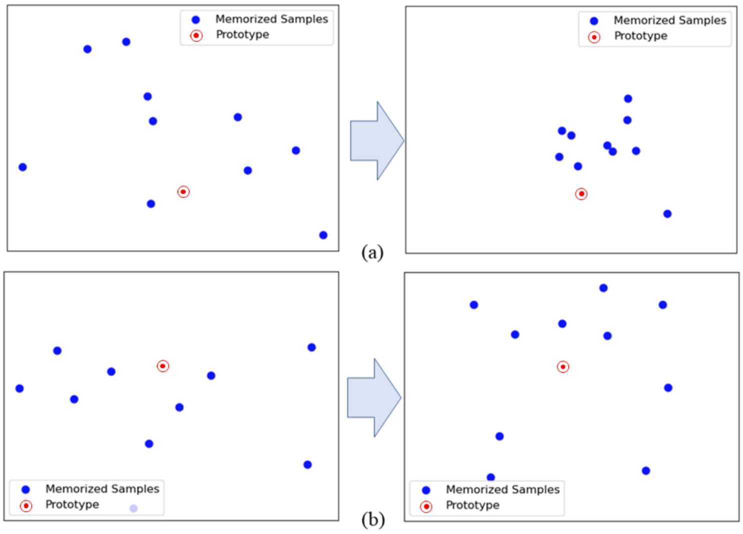


Figure 3: Visualized process of alleviating the disruption of sample embedding space after learning a new task. (a) Recovery result of EMAR+BERT. (b) Recovery result of RP-CRE.

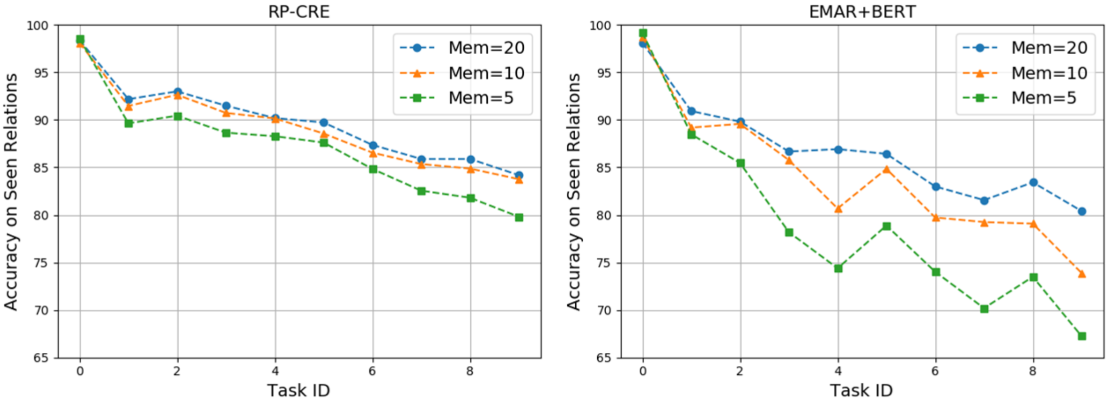
4.5 Long-term Effectiveness of Episodic Memory

To explore long-term effectiveness of episodic memory in our model, we compared our model with EMAR+BERT on FewRel, which is similar to our model in selecting memorized samples. Results are shown in Table 2, where each score is the classification accuracy for all relations on test set of each former task. We conclude that after training on 10 sequential tasks, our model performs better on the former tasks. It indicates that our model has a much stable understanding of old relations in old tasks. In both models, memorized samples of old relations are used to restore the model's performance on old relations (memory reconsolidation in EMAR, prototypical refining in our model). In order to find the reason of EMAR's inferior performance on the former tasks, we display the visualization the varying of sample embedding space during model training.

Concretely, we used *t-SNE* (Van der Maaten and Hinton, 2008) for dimension reduction and chose memorized samples from relation *participant* for visualization, which were fed into the two models on the same task. Figure 3 shows the sample positions in the embedding space, where the blue dots represent the memorized samples and the red dot represents the relation's prototype (the centroid of memorized samples before learning the new task). Left two sub-figures display how sample embedding space is disrupted by the learning of new tasks. Right two sub-figures display how the model recovers.

From Figure 3, we notice that although EMAR's sample embeddings are getting closer to the former centroid (relation prototype) after memory reconsolidation, they converge in fact. Comparatively, our model restores the embedding space while retaining the diversity between samples. In terms of typicality and diversity of memorized samples, it is not our purpose to encode all memorized samples into exactly the same point in the embedding space, since it may damage the diversity of these samples and reduce the information provided by the samples, during model's recovery from disrupted condition.

This result is mainly due to that the loss function selected in EMAR's memory reconsolidation is too radical, focusing only on reducing the absolute distance between a memorized sample and the relation prototype. Therefore, our strategy of refining sample embeddings with relation prototypes (Section



(a) (b)

Figure 4: Comparison of model's dependence on memory size, indicating that our model has a weaker dependence on memory size. The X-axis is the serial ID of current task, and Y-axis represents the model's classification accuracy on test set from all observed relations at current stage.

3.6) better preserves the diversity of memorized samples, as it takes into account various features of samples rather than the true labels. It eventually retains more information of memorized samples.

4.6 Model Dependence on Memory Size

In most memory-based CRE models, memory size (number of memorized samples) is a key factor affecting model performance. However, most of previous models do not fully utilize the information provided by memorized samples, resulting in their dependence on memory size. Even worse, the memorized samples have the same magnitude as the original samples. In Section 3.6, we have emphasized the advantages of our model in retaining and full utilization of memory information. We verified whether our model relies less on memory size through comparison experiments, of which the results are shown in Figure 4.

We chose EMAR+BERT as the main competitor, in which the configuration and task sequence remained unchanged. The only variable we adjusted is memory size. Based on the results we conclude that, as memory size decreases, our model obtains less decreased performance than EMAR+BERT (performance degradation is inevitable). Even though EMAR showed a relatively stable performance in the first two tasks, its performance dropped significantly in the subsequent tasks. This is consistent with the long-term effectiveness of memory we have analyzed in Section 4.5. The diversity of samples in EMAR would gradually disappear, making it highly dependent on memory size. Comparatively, our model's dependence on memory size is weak because it preserves the diversity of samples.

5 Conclusion

In this paper, we propose a novel CRE model obtaining enhanced performance through refining sample embeddings. In our model, the sample embeddings are refined by an attention-based memory network fed with relation prototypes, that are generated from memorized samples. The comparison experiments show that our model significantly outperforms current state-of-the-art CRE models. As most current CRE models are memory-based, we further explore the long-term effectiveness of episodic memory. The results show that our model has great advantages in maintaining diversity of memorized samples and performs well in avoiding catastrophic forgetting of old relations (tasks). Because of the efficiency in memory mechanism, our model depends less on memory size. In future work, we will explore whether the mechanism of refining sample embeddings with prototypes can be used in other classification-based continual learning tasks.

Acknowledgments

We sincerely thank all anonymous reviewers for their valuable comments. This work is supported by National Key Research and Development Project (No.2020AAA0109302), Shanghai Science and Technology Innovation Action Plan (No.19511120400) and Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision*, pages 139–154.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2018. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*.

Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. 2015. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Ning Ding, Xiaobin Wang, Yao Fu, Guangwei Xu, Rui Wang, Pengjun Xie, Ying Shen, Fei Huang, Hai-Tao Zheng, and Rui Zhang. 2021. Prototypical representation learning for relation extraction. *arXiv preprint arXiv:2103.11647*.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1409–1418.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. 2020. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. 2017. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258.

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

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2017. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

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*, pages 2124–2133.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. 2018. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition*, pages 2262–2268.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. *arXiv preprint arXiv:1706.08840*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48.

Abiola Obamuyide and Andreas Vlachos. 2019. Meta-learning improves lifelong relation extraction. In *Proceedings of the 4th Workshop on Representation Learning for NLP*, pages 224–229.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, pages 1842–1850.

Adam Santoro, Ryan Faulkner, David Raposo, Jack Rae, Mike Chrzanowski, Theophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy Lillicrap. 2018. Relational recurrent neural networks. *arXiv preprint arXiv:1806.01822*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090.

Sebastian Thrun and Tom M Mitchell. 1995. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 796–806.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1476–1488.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2361–2364.

Tongtong Wu, Xuekai Li, Yuan-Fang Li, Reza Haffari, Guilin Qi, Yujin Zhu, and Guoqiang Xu. 2021. Curriculum-meta learning for order-robust continual relation extraction. *arXiv preprint arXiv:2101.01926*.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1785–1794.

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.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995.

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

Runyan Zhang, Fanrong Meng, Yong Zhou, and Bing Liu. 2018a. Relation classification via recurrent neural network with attention and tensor layers. *Big Data Mining and Analytics*, 1(3):234–244.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018b. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 207–212.

Hao Zhu, Yankai Lin, Zhiyuan Liu, Jie Fu, Tat-Seng Chua, and Maosong Sun. 2019. Graph neural networks with generated parameters for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1331–1339.