Consistent Prototype Learning for Few-Shot Continual Relation Extraction

Xiudi Chen[1,2,*] **Hui Wu**[1,2,*] **Xiaodong Shi**[1,2,3,†]
[1]Department of Artificial Intelligence, School of Informatics, Xiamen University, China
[2]National Institute for Data Science in Health and Medicine, Xiamen University, China
[3]Key Laboratory of Digital Protection and Intelligent Processing of Intangible Cultural Heritage of Fujian and Taiwan (Xiamen University), Ministry of Culture and Tourism, China
{chenxiudi,huistudent}@stu.xmu.edu.cn, mandel@xmu.edu.cn

Abstract

Few-shot continual relation extraction aims to continually train a model on incrementally few-shot data to learn new relations while avoiding forgetting old ones. However, current memory-based methods are prone to overfitting memory samples, resulting in insufficient activation of old relations and limited ability to handle the confusion of similar classes. In this paper, we design a new **N**-way-**K**-shot **C**ontinual **R**elation **E**xtraction (NK-CRE) task and propose a novel few-shot continual relation extraction method with **Con**sistent **P**rototype **L**earning (ConPL) to address the aforementioned issues. Our proposed ConPL is mainly composed of three modules: 1) *a prototype-based classification module* that provides primary relation predictions under few-shot continual learning; 2) *a memory-enhanced module* designed to select vital samples and refined prototypical representations as a novel multi-information episodic memory; 3) *a consistent learning module* to reduce catastrophic forgetting by enforcing distribution consistency. To effectively mitigate catastrophic forgetting, ConPL ensures that the samples and prototypes in the episodic memory remain consistent in terms of classification and distribution. Additionally, ConPL uses prompt learning to extract better representations and adopts a focal loss to alleviate the confusion of similar classes. Experimental results on two commonly-used datasets show that our model consistently outperforms other competitive baselines[1].

1 Introduction

Continual relation extraction (CRE) aims to continually train a model on new data to learn new relations while avoiding forgetting old relations. Due to its wide applicability to real-world applications, CRE has attracted increasing attention (Obamuyide and Vlachos, 2019; Han et al., 2020; Zhao et al., 2022). Most existing studies adopt memory-based approaches as the principal architecture and achieve great success in CRE. For example, Cui et al. (2021) employ relation prototype to extract useful information of each relation. Wang et al. (2022) adopt a simple yet effective adversarial class augmentation mechanism to learn more precise and robust representations. However, obtaining large labeled data can be time-consuming and expensive. Qin and Joty (2022) introduce continual few-shot relation learning (CFRL) and propose a novel method of embedding space regularization and data augmentation. But, the first task in the CFRL setting still involves a substantial amount of training data, and the evaluation results obtained in this setting may not completely reflect the performance of the model under true continual few-shot

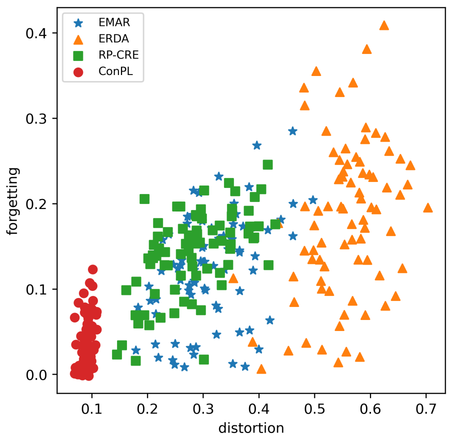


Figure 1: The scatter plot that reflects the relationship between distortion and forgetting in four methods, namely EMAR (Han et al., 2020), RP-CRE (Cui et al., 2021), ERDA (Qin and Joty, 2022) and our proposed ConPL. The abscissa “distortion” indicates the degree of change in feature embedding, and the ordinate “forgetting” indicates the degree of forgetting. Each point on the plot represents a prototype of a relation class. The detailed calculations are shown in Appendix A.1.

*Equal contribution.
†Correspondence author.
[1]https://github.com/XiudiChen/ConPL

learning conditions[2]. This is because a portion of the test set is drawn from the first task, which can lead to a big gap between the results obtained in the CFRL setting and those obtained under true continual few-shot relation learning conditions.

To this end, we propose the **N**-way-**K**-shot **C**ontinual **R**elation **E**xtraction (NK-CRE), which strictly adheres to the N-way K-shot few-shot setting and all classes contain only a small amount of labeled instances. In the NK-CRE setting, we employ existing CRE methods to conduct a series of experimental analyses for few-shot continual relation learning, as illustrated in Figure 1. We observe that existing CRE methods, particularly ERDA, exhibit significant distortion and forgetting. Additionally, we find that the prototype distortion is highly relevant to forgetting in NK-CRE. Based on these findings, we hypothesize that reducing the prototype distortion can greatly mitigate catastrophic forgetting in NK-CRE. However, existing CRE methods in the NK-CRE setting face the challenge of overfitting the memory samples due to the limited memory available, leading to insufficient activation of the old relations and resulting in significant distortion and forgetting.

Besides, the confusion of similar classes is also a major cause of catastrophic forgetting in CRE (Wang et al., 2022), which is particularly acute in NK-CRE. For instance, the prototype embeddings of similar classes, like "father" and "mother", tend to be highly similar due to their similar context and entity pairs. When "father" has already been learned in a previous task and "mother" is shown in a new task, the model prioritizes learning the representation of the current class "mother", but may disregard the subtle differences between "mother" and "father", leading to catastrophic forgetting.

To deal with the above issues, we propose a novel **Con**sistent **P**rototype **L**earning (ConPL) method for few-shot continual relation extraction in order to effectively activate old relations and mitigate catastrophic forgetting. ConPL consists of three primary modules: 1) a prototype-based classification module, which leverages prompt learning to extract better relation representations and utilizes temporary prototypes constructed from new task data and several distance metric-based losses for basic classification; 2) a memory-enhanced module that comprises sample memory for storing vital labeled samples and prototype memory for storing optimal prototype representations; 3) a consistent learning module that addresses the problem of unbalanced distribution of new and old relations through consistency learning between sample distribution. In addition, we introduce a focal loss to mitigate the impact of the confusion of similar classes on catastrophic forgetting. This loss function allows the model to focus on the training of similar classes, enabling it to pay more attention to the distinctions between similar classes and reducing the impact of confusion.

In this study, experimental results of FewRel and TACRED datasets in the NK-CRE setting demonstrate that ConPL outperforms existing models by a significant margin. Further, ablation experiments show that each component of our method contributes to its effectiveness. Our analysis also reveals that ConPL not only significantly reduces the distortion of class prototypes, but also effectively minimizes catastrophic forgetting. Overall, our ConPL offers a robust and effective approach to few-shot continual relation extraction.

2 Related Work

Continual Learning, also known as lifelong learning and incremental learning, is the process of training models to perform well on task streams with different data distributions. Its main methods can be divided into three categories: (i) regularization-based methods (Kirkpatrick et al., 2017; Zenke et al., 2017); (ii) architecture-based methods (Fernando et al., 2017; Wortsman et al., 2020); and (iii) memory-based methods (Rolnick et al., 2019; Cha et al., 2021). Memory-based methods, also called rehearsal-based methods, have been shown to be particularly suitable for natural language processing (NLP), especially for CRE (Wang et al., 2019; Han et al., 2020; Cui et al., 2021; Zhao et al., 2022). These methods selectively store old samples in limited memory, training them with new samples to prevent catastrophic forgetting. For example, EMAR (Han et al., 2020) utilizes relation prototypes for memory reconsolidation exercises. RP-CRE (Cui et al., 2021) uses prototypes of all observed relations to refine subsequent sample embeddings. CRECL (Hu et al., 2022) builds a classification network and a prototypical contrastive network. However, these methods only compute

[2]True continual few-shot learning, also known as few-shot continual learning, refers to a model learning to perform few-shot learning on a continuous stream of tasks, with each task typically comprising a small number of labeled examples.

temporary prototypes before training a new task and store memory samples, which can lead to significant distortion and forgetting as the number of tasks increases. In contrast, our approach stores prototypes in memory without modification.

Few-shot Learning aims to address the overfitting issue that may arise when there is a limited amount of labeled training data for model training. Existing methods can be divided into two categories: (i) data-based (Tsai and Salakhutdinov, 2017; Benaim and Wolf, 2018); and (ii) meta learning-based, including optimization-based algorithm (Finn et al., 2017) and metric-based algorithm. Among them, prototypical network (Snell et al., 2017) is a simple yet efficient metric-based approach and has become the most mainstream method in few-shot relation extraction (Yang et al., 2020; Qu et al., 2020; Han et al., 2021a). In this paper, our method is built on the prototypical network and proposes storing class prototypes in memory.

Prompt Learning is a simple yet effective method for fine-tuning pre-trained language models (PLMs), which uses prompt information to enrich the input and enable better mining of prior knowledge in PLMs. According to the pre-training objectives, the structure of prompts is mainly divided into two types: (i) adding the task description before the input, e.g., the generative model GPT-3 (Brown et al., 2020); and (ii) formalizing downstream tasks into cloze-style tasks, e.g., the discriminative model BERT (Devlin et al., 2018). According to different forms of expression, prompt can be divided into continuous (Lester et al., 2021; Vu et al., 2022) and discrete (Schick et al., 2020; Han et al., 2021a). Continuous prompts are composed of vectors and the optimal prompt embedding can be learned automatically by the model, while discrete prompts are manually constructed by people based on their experience. For relation extraction, several works (Han et al., 2021b; Chen et al., 2022) have investigated the utilization of the prior knowledge of PLMs through prompt learning, which has been shown to be effective in few-shot learning. In this paper, we use discrete prompts with cloze-style to extract better representations in few-shot CRE.

3 Problem Formulation

N-way-**K**-shot **C**ontinual **R**elation **E**xtraction (NK-CRE) is a continual relation learning task in the N-way K-shot setting scenario, which aims to continually train the model on new tasks to learn new relations while avoiding forgetting previously learned ones under the few-shot scenario. Compared to CFRL (Qin and Joty, 2022), NK-CRE strictly adheres to the N-way K-shot setting, which is more challenging yet more representative of real-world applications.

In NK-CRE, the model is trained on a sequence of tasks $\{\mathcal{T}^1, \mathcal{T}^2, \ldots, \mathcal{T}^n\}$, where each task $\mathcal{T}^k (k \in [1, \ldots, n])$ has its own training set D^k_{train}, test set D^k_{test} and corresponding relation set R^k. Each dataset $D = \{(x_i, y_i)\}_{i=1}^{N \times K}$ contains N classes and K samples of each class, where each sample (x_i, y_i) consists of a sentence x_i with a pair of entities (e_h, e_t) and a relation label $y_i \in R^k$ of the entity pair, and N is the number of current relations in R^k. After being trained on $\mathcal{T}^k$ at the k-th task to learn the new relations R^k, the model will be evaluated on the test sets of previous k tasks $\hat{D}^k_{\text{test}} = \cup_{i=1}^k D^i_{\text{test}}$ to verify the discriminant ability of the model for all known relations $\hat{R}^k = \cup_{i=1}^k R^i$.

Different from the memory used in previous works (Han et al., 2020; Cui et al., 2021; Qin and Joty, 2022), we propose a novel multi-information episodic memory $\mathcal{M} = \{\mathcal{M}^1, \mathcal{M}^2, \ldots\}$ to alleviate catastrophic forgetting in continual learning, where $\mathcal{M}^k$ consists of the most informative samples S^k and prototype representations P^k corresponding to relations R^k in $\mathcal{T}^k$. Although each relation in the memory can store multiple samples, we store a sample per relation under a limited capacity. After the k-th task is trained, the memory $\mathcal{M}$ contains sample set $\hat{S}^k = \cup_{i=1}^k S^i$ and prototype set $\hat{P}^k = \cup_{i=1}^k P^i$ of the previous k tasks.

4 Methodology

In this section, we elaborate on our proposed method ConPL, which is composed of three modules: a prototype-based classification module (Section 4.1), a memory-enhanced module (Section 4.2), and a consistent learning module (Section 4.3). Prototype-based classification module is the basic component of ConPL. Memory-enhanced module introduces a novel multi-information episodic memory equipped with some samples and corresponding prototypical representations. Consistency learning module adds an extended training process to balance the predictions of all known relations. Finally, we give a detailed description of the overall training procedure (Section 4.4).

4.1 Prototype-Based Classification Module

Encoder. Given a sentence x with a head entity e_h and a tail entity e_t, we choose BERT (Devlin et al., 2018) as the encoder with parameters θ to obtain the relational representation of entity pairs in x. Specifically, we firstly enrich the sentence x into a specific input sequence $x_{\text{input}} = \{[\text{CLS}], e_h, [\text{MASK}], e_t, [\text{SEP}], x, [\text{SEP}]\}$ based on the prompt learning (Han et al., 2021b; Chen et al., 2022), and then encode the input sequence by the encoder to obtain the contextualized representation. Here we use the vector representation of the special token $[\text{MASK}]$ as the relational representation.

$$\boldsymbol{h}_{[\text{MASK}]} = f_\theta\left(x_{\text{input}}\right) \tag{1}$$

For simplicity, we will use x to refer to the specific input sequence x_{input}.

Initializing temporary prototypes of new classes. When firstly training on the k-th new task to learn new relations, we use all samples of each new relation in the current task to construct prototypical representations for the corresponding classes. Specifically, we encode all samples in D_{train}^k, and then use an aggregation operator (such as averaging) to aggregate the embeddings of samples from the same class to calculate the prototypical representation $\mathbf{p}_j$ of each class.

$$\mathbf{p}_j = \frac{1}{|D_j^k|} \sum_{(x_i,y_i)\in D_j^k} f_\theta\left(x_i\right) \tag{2}$$

where $D_j^k = \left\{(x_i, y_i) \mid (x_i, y_i) \in D_{train}^k, y_i = r_j\right\}$, $|D_j^k|$ is the number of samples and $\mathbf{p}_j$ represents the prototypical representation of $r_j (r_j \in R^k)$ in the new task. So the temporary prototypes of current new task is $\tilde{P}^k = \bigcup_{r_j\in R^k} \mathbf{p}_j$. Based on previous relation prototypes $\hat{P}^{k-1}$ of the memory, we can get all current prototypes $\bar{P}^k = \hat{P}^{k-1} \cup \tilde{P}^k$ [3] related to all classes we have seen.

Prototype classifier with experience replay. To learn new relations while retaining existing relation knowledge, we use the experience replay to train the model on the new train dataset $\bar{D}_{train}^k = D_{train}^k \cup \hat{S}^{k-1}$, which combine the training samples of the k-th task with the samples of the previous $k-1$ tasks in the memory. The relation distribution for each sample x_i is computed as:

$$p(r_i|x_i) = \frac{\exp\left(d\left(f_\theta\left(x_i\right), \mathbf{p}_i\right)\right)}{\sum_{l=1}^{|\hat{R}^k|} \exp\left(d\left(f_\theta\left(x_i\right), \mathbf{p}_l\right)\right)} \tag{3}$$

where $d\left(.,.\right)$ represents the distance measurement formula by the cosine similarity, $\mathbf{p}_l$ is the prototypical representation of $r_l (r_l \in \hat{R}^k)$ in $\bar{P}^k$, and $|\hat{R}^k|$ is the number of all known relations in previous k tasks.

The **cross entropy loss** $\mathcal{L}_{ce}$ for classification is calculated in a distance measurement way.

$$\mathcal{L}_{ce} = -\sum_{(x_i,y_i)\in \bar{D}_{train}^k} \log p(r_i|x_i) \tag{4}$$

In order to wake up the old relational knowledge, we propose the **classification consistency loss** $\mathcal{L}_{cc}$ to focus on the consistent correlation between the feature of each sample in $\hat{S}^{k-1}$ and the corresponding prototypical representation in $\hat{P}^{k-1}$ based on current memory.

$$\mathcal{L}_{cc} = \sum_{(x_i,y_i)\in \hat{S}^{k-1}} \|f_\theta(x_i) - \mathbf{p}_i\| \tag{5}$$

Considering the different correlation between the new relations and previous relations (e.g., the relation "*mother*" in the new task and previous relations "*father*" / "*spouse*" belongs to greatly similar relations.), we select similar classes to highlight their distinction. Specifically, we start by obtaining the most similar negative prototype $\mathbf{p}_i^{mn} = \text{argmax}(d\left(f_\theta\left(x_i\right), \mathbf{p}_s\right))$ for each sample x_i, which is the prototype of the most likely relation of miscalculation. Next, we set a threshold α and automatically screen some confusing negative prototypes $\hat{\mathbf{p}}_i^{sn} = \{\mathbf{p}_i^{sn} \mid d(f_\theta(x_i), \mathbf{p}_i) - d(f_\theta(x_i), \mathbf{p}_i^{sn}) < \alpha\}$ that include prototypes that are the most similar to the prototype $\mathbf{p}_i$ with the target relation. Here $\mathbf{p}_s$ and $\mathbf{p}_i^{sn}$ refer to any prototype other than $\mathbf{p}_i$, so $\mathbf{p}_s \neq \mathbf{p}_i$ and $\mathbf{p}_i^{sn} \neq \mathbf{p}_i$. The distribution of x_i is computed for distinguishing similar classes.

$$p_s(r_i|x_i) = \frac{\exp\left(d\left(f_\theta\left(x_i\right), \mathbf{p}_i\right)\right)}{\sum_{l=1}^{|P_i^{sim}|} \exp\left(d\left(f_\theta\left(x_i\right), \mathbf{p}_l\right)\right)} \tag{6}$$

where $P_i^{sim} = [\mathbf{p}_i; \mathbf{p}_i^{mn}; \hat{\mathbf{p}}_i^{sn}]$ represents the set of prototypes similar to the output feature of x_i and $|P_i^{sim}|$ is the number.

[3] In this paper, $\tilde{P}^k$ refers to the temporary prototypes of the k-th task and P^k refers to the prototypes of the k-th task stored in memory. $\bar{P}^k$ refers to the prototypes of the previous k task that contain the prototypes from memory for the previous $k-1$ tasks as well as the temporary prototypes for the k-th task. Additionally, $\hat{P}^k$ refers to the prototypes of the previous k task stored in memory.

Therefore, we adopt the **focal loss** to alleviate the difficulty in predicting similar classes.

$$\mathcal{L}_{fc} = -\sum_{(x_i,y_i)\in\bar{D}^k_{train}} \log p_s(r_i|x_i) \tag{7}$$

4.2 Memory-Enhanced Module

Compared to previous memory-based methods (Han et al., 2020; Cui et al., 2021; Qin and Joty, 2022), we add prototypical representations into the episodic memory for enriching the old relations. So the memory of our model is separated into two parts: the *Sample Memory* $\hat{S}^k$, which is used to store samples with class labels, and the *Prototype Memory* $\hat{P}^k$, which is used to store feature embeddings of class prototypes.

For the sample memory, we calculate the prototypical representation of each relation in the current task $\mathcal{T}^k$ based on Eq (2), and then select some closest samples as the typical samples to store them in S^k, where each relation of $\mathcal{T}^k$ only stores one typical sample. During training, prototypical representations computed by all relation samples can be unstable. Therefore, once the typical samples are determined, we use the feature representations of the selected typical samples to update prototypical representations of the current task and store their feature representations in prototype memory P^k after $\mathcal{T}^k$ is trained.

4.3 Consistent Learning Module

Considering the unbalanced distribution of new relations and old ones in the training dataset $\bar{D}^k_{train}$, we add an extended training step that solely focuses on the memory $\mathcal{M}$ that helps to balance the learning of all known relations and distinguish new and old relations. Therefore, we propose the **distribution consistency loss** $\mathcal{L}_{dc}$ via computing the consistent constraint between the sample distribution and the prototype distribution.

$$\mathcal{L}_{dc} = \sum_{(x_i,y_i)\in\hat{S}^k} \left\| d\left(f_\theta(x_i), \hat{P}^k\right) - d\left(\mathbf{p}_i, \hat{P}^k\right) \right\| \tag{8}$$

4.4 Learning Procedure

Joint training objective. We adopt a three-stage training strategy to train our model. The overall training objective of the first two stages is computed as follows.

$$\mathcal{L}_{class} = \lambda_{ce}\mathcal{L}_{ce} + \lambda_{cc}\mathcal{L}_{cc} + \lambda_{fc}\mathcal{L}_{fc} \tag{9}$$

Algorithm 1 Training Process for the k-th task.

Input: The training set D^k_{train} and the new relation set R^k; the sample memory $\hat{S}^{k-1} = \cup_{i=1}^{k-1} S^i$ and the prototype memory $\hat{P}^{k-1} = \cup_{i=1}^{k-1} P^i$ of previous $k-1$ tasks; all known relations $\hat{R}^{k-1} = \cup_{i=1}^{k-1} R^i$ of previous $k-1$ tasks; the values of epoch$_1$, epoch$_2$ and epoch$_3$.
Output: Configurations of the encoder with parameters θ; the sample memory S^k and the prototype memory P^k of the k-th task.
Initialize: The encoder with parameters θ.

```
1:  INITIALIZE temporary prototype $\tilde{P}^k$ of each relation $r_i \in R^k$ based on $D^k_{train}$
2:  $\hat{R}^k \leftarrow \hat{R}^{k-1} \cup R^k$
3:  $\bar{P}^k \leftarrow \hat{P}^{k-1} \cup \tilde{P}^k$
4:  $\bar{D}^k_{train} \leftarrow D^k_{train} \cup \hat{S}^{k-1}$
5:  for $i \in \{1, \cdots, \text{epoch}_1\}$ do
6:      UPDATE $\theta$ with $\mathcal{L}_{class}$ on $\bar{D}^k_{train}$ and $\hat{S}^{k-1}$
7:      UPDATE $\tilde{P}^k$ in $\bar{P}^k$
8:  end for
9:  SELECT a key sample of each relation from $D^k_{train}$ to store into $S^k$
10: REINITIALIZE prototype $P^k$ of each relation $r_i \in R^k$ based on $S^k$
11: $\hat{S}^k \leftarrow \hat{S}^{k-1} \cup S^k$
12: $\bar{P}^k \leftarrow \hat{P}^{k-1} \cup P^k$
13: $\bar{D}^k_{train} \leftarrow D^k_{train} \cup \hat{S}^k$
14: for $i \in \{1, \cdots, \text{epoch}_2\}$ do
15:     UPDATE $\theta$ with $\mathcal{L}_{class}$ on $\bar{D}^k_{train}$ and $\hat{S}^k$
16:     UPDATE $P^k$ in $\bar{P}^k$
17: end for
18: for $i \in \{1, \cdots, \text{epoch}_3\}$ do
19:     UPDATE $\theta$ with $\mathcal{L}_{cons}$ on $\hat{S}^k$
20:     UPDATE $P^k$ in $\bar{P}^k$
21: end for
22: $\hat{P}^k \leftarrow \hat{P}^{k-1} \cup P^k$
```

And the overall training objective of the third stage is defined as follows.

$$\mathcal{L}_{cons} = \lambda_{ce}\mathcal{L}_{ce} + \lambda_{cc}\mathcal{L}_{cc} + \lambda_{fc}\mathcal{L}_{fc} + \lambda_{dc}\mathcal{L}_{dc} \tag{10}$$

where λ_{ce}, λ_{cc}, λ_{fc} and λ_{dc} are the relative weights of the component losses, respectively. It is noteworthy that the training set of the different stages is different. The first stage adopts the sample memory $\hat{S}^{k-1}$. After storing the sample memory S^k of the k-th task, the second and third stages use $\hat{S}^k$ for training.

Training procedure. The overall training procedure is summarized in Algorithm 1. In the first stage, we use all samples of each class in D^k_{train} to obtain the prototype representation $\tilde{P}^k$ of all classes in R^k. And then based on all prototype $\bar{P}^k = \hat{P}^{k-1} \cup \tilde{P}^k$, the sample memory $\hat{S}^{k-1}$, and the current training set $\bar{D}^k_{train} = D^k_{train} \cup \hat{S}^{k-1}$, we let the model learn the knowledge of the new classes (line 1-8). In the second stage, we firstly select key samples S^k closest to the center of each class and store them in the sample memory $\hat{S}^k = \hat{S}^{k-1} \cup S^k$, and use S^k to construct new prototype P^k of the new classes for subsequently training with $\bar{P}^k = \hat{P}^{k-1} \cup P^k$ (line 9-17). In the third stage, we only use $\hat{S}^k$ to predict all known relations and further ensure that the output of the samples in memory is consistent with their stored features (line 18-22).

5 Experiments

5.1 Datasets

In line with previous work (Qin and Joty, 2022), our experiments will be conducted on two commonly used datasets, **FewRel** (Han et al., 2018) and **TACRED** (Zhang et al., 2017). For NK-CRE, we evaluate our method under three different few-shot settings of FewRel, namely 10-way-2-shot, 10-way-5-shot, and 10-way-10-shot, and two different few-shot settings of TACRED, namely 5-way-5-shot and 5-way-10-shot. The detailed introduction of two datasets is shown in Appendix A.2.

5.2 Evaluation Methods

After training the model on the k-th task, we use the **whole accuracy** metric to evaluate the model's performance on the union of the test sets of previous k tasks $\hat{D}^k_{\text{test}} = \cup^k_{i=1} D^i_{\text{test}}$. This metric is used to evaluate the model's ability to alleviate catastrophic forgetting while acquiring new knowledge with a few-shot new task.

In addition, we also measure the degree of catastrophic forgetting using the **forgetting** metric proposed by Chaudhry et al. (2018). The forgetting metric measures the degree to which the model has forgotten previously learned knowledge after training on new tasks. Specifically, after training on all the tasks, the authors measure the model's performance on the test sets of all the previous tasks. The specific formula for forgetting is as follows:

$$F_k = \frac{1}{n-k} \sum_{j=k+1}^{n} g^j_k \tag{11}$$

$$g^j_k = \max_{l \in \{k, \cdots, j-1\}} a_{l,k} - a_{j,k} \tag{12}$$

where $F_k(k \in [1, \ldots, n-1])$ represents the forgetting of the k-th task after all tasks have been trained, $a_{j,k}(j > k)$ is the accuracy of the model on the k-th task after the j-th task training is completed, and g^j_k is the forgetting of the k-th task after the j-th task training is completed.

Considering that the order of task sequences will affect the final performance of the model, we randomly generate 6 task sequences for experiments and calculate their average accuracy as the final result. To ensure a fair comparison between the baselines and the proposed method, we set the same random seed for the baselines and our proposed method to ensure that they are evaluated on the same set of task sequences.

5.3 Baselines

We compare our proposed method with the following baselines:

EMAR (Han et al., 2020): A classical method for continual relation learning, which uses relation prototypes for memory reconsolidation exercise to keep a stable understanding of the old relations.

RP-CRE (Cui et al., 2021): A method for continual relation learning, where prototype embeddings are computed based on memorized samples and are used to re-initialize a memory network to refine subsequent sample embeddings.

ERDA (Qin and Joty, 2022): A SOTA method for continual few-shot relation learning (CFRL) that employs embedding space regularization and data augmentation to improve model performance.

Considering that PLMs (e.g., BERT) have been shown to outperform Bi-LSTM on many NLP tasks, we replace the Bi-LSTM with BERT to reproduce the baselines we mentioned. Furthermore, recent advances in Prompt Learning have shown great performance on many NLP tasks, by enabling models to learn from prompt-based examples and generate coherent outputs. Therefore, we construct prompt templates at the input to generate their variants **EMAR(PT)**, **RP-CRE(PT)**, and **ERDA(PT)** for better comparison.

5.4 Training Details

During training, we use $\text{BERT}_{\text{BASE}}$ (Devlin et al., 2018) as the encoder and Adam optimizer with a learning rate of $2e^{-5}$ for gradient updates. To prevent gradient overflow, we set the gradient clipping value to 10. The loss weights λ_{ce}, λ_{cc}, λ_{fc} and λ_{dc} are all set to 1.0, and α is set to 0.1. In Algorithm 1, epoch_1 and epoch_2 are set to 1, and epoch_3 is set to 3.

5.5 Main Results

Table 1 presents the whole accuracy of the FewRel benchmark in the 10-way-5-shot setting and the TACRED benchmark in the 5-way-5-shot setting.

Method	Task Index							
	T1	T2	T3	T4	T5	T6	T7	T8
10-way-5-shot of FewRel								
EMR (Wang et al., 2019)	96.35	88.02	78.83	75.15	72.00	69.41	66.70	63.68
EMAR (Han et al., 2020)	92.03	78.87	72.81	69.19	68.05	66.23	63.68	61.77
IDLVQ-C (Chen and Lee, 2021)	96.03	87.18	76.63	73.57	67.74	65.16	62.64	60.32
ERDA (Qin and Joty, 2022)	96.38	88.91	83.10	79.73	74.83	72.84	70.28	68.07
EMAR† (Han et al., 2020)	82.50	76.23	71.41	66.93	62.02	62.37	60.05	59.29
RP-CRE† (Cui et al., 2021)	86.38	78.47	71.93	69.07	66.28	65.88	62.28	61.17
ERDA† (Qin and Joty, 2022)	88.15	79.65	74.05	68.18	67.14	66.96	65.86	63.60
EMAR(PT)† (Han et al., 2020)	95.28	92.75	90.57	88.15	87.04	84.90	83.07	81.34
RP-CRE(PT)† (Cui et al., 2021)	93.52	89.48	87.67	85.28	84.48	83.08	81.92	80.87
ERDA(PT)† (Qin and Joty, 2022)	**96.55**	92.56	88.56	84.47	84.14	79.94	78.45	77.02
ConPL(Ours)	95.72	**93.53**	**91.31**	**89.95**	**88.93**	**88.39**	**87.43**	**85.77**
5-way-5-shot of TACRED								
EMAR† (Han et al., 2020)	68.71	51.53	43.86	38.54	34.08	32.06	29.9	27.87
RP-CRE† (Cui et al., 2021)	74.42	51.14	40.99	32.43	30.82	26.62	25.82	24.12
ERDA† (Qin and Joty, 2022)	77.55	56.31	47.66	39.72	36.71	33.46	31.07	28.50
EMAR(PT)† (Han et al., 2020)	94.88	86.36	83.25	80.60	77.76	75.43	73.85	68.67
RP-CRE(PT)† (Cui et al., 2021)	90.98	84.71	78.96	75.64	73.07	71.14	66.99	65.31
ERDA(PT)† (Qin and Joty, 2022)	95.77	86.55	78.59	74.58	69.31	66.53	61.92	55.97
ConPL(Ours)	**97.34**	**89.85**	**86.33**	**82.53**	**81.21**	**79.56**	**78.38**	**76.38**

Table 1: Whole accuracy (%) of different methods after training for each task on the **FewRel** benchmark in the **10-way-5-shot** setting and on the **TACRED** benchmark in the **5-way-5-shot** setting. † denotes our reproduced results with the publicly available codebases, where PT represents using prompt learning to enrich the inputs. Other results are obtained directly from Qin and Joty (2022) in the CFRL setting, which has enough training data (100 samples per relation) in the first task. The best values are denoted in **bold**.

Based on the main results of NK-CRE, we can observe that:

(1) **Prompt learning can obtain better semantic representations and improve model performance in few-shot scenarios.** Baselines with PT can achieve the desired effect directly on each task in NK-CRE, particularly for the first task, without the need for training on a large amount of labeled data like CFRL (Qin and Joty, 2022). On the FewRel's 10-way-5-shot, EMAR(PT) is 12.78% higher than EMAR after learning the first task using the 5-shot setting and 3.25% higher than EMAR after learning the first task with a large amount of labeled data. Notably, our method ConPL achieves the highest performances in most of the tasks. In the first task, our method ConPL achieves the accuracy of 95.72%, which was slightly lower than the accuracy achieved by ERDA(PT) at 96.55%. However, it is worth noting that ERDA(PT) leverages external Wikipedia sentences for data augmentation, which may account for its slightly superior performance. On the TACRED's 5-way-5-shot, ConPL achieves the highest performance on all tasks.

(2) **Consistent prototype learning with multi-information memory can significantly improve model performance in NK-CRE.** When faced with multiple tasks (T2∼T8), Our method ConPL significantly outperforms all existing methods on FewRel's 10-way-5-shot and TACRED's 5-way-5-shot. Compared with all baselines, ConPL obtains the highest improvement reaching 26.48% of FewRel's 10-way-5-shot and 41.19% of TACRED's 5-way-5-shot after learning all tasks. More specifically, the performance obtained by our ConPL at T8 is better than that of EMAR(PT) at T6, RP-CRE(PT) and ERDA(PT) at T4. It proves that our consistent prototype learning strategy is highly effective.

Moreover, we present more detailed results in Table 1 in Appendix A.3, including the means and variances of all sequential results. In T1∼T8 of FewRel's 10-way-5-shot, the mean deviations of MEAR(PT), RP-CRE(PT), ERDA(PT), and our ConPL are 2.76%, 1.92%, 5.27%, and 1.50% respectively, which indicate that our method ConPL is more stable than other baselines. In addition, Figure 2 presents the results of FewRel's 10-way-

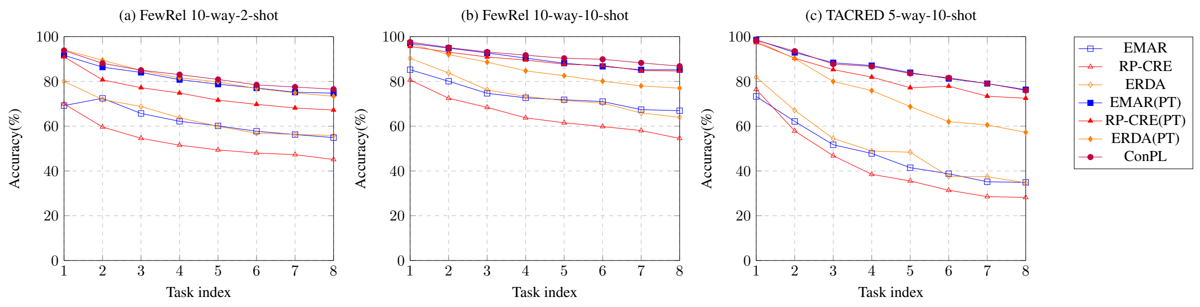


Figure 2: Whole accuracy (%) of different methods after training for each task on FewRel's 10-way-2-shot, FewRel's 10-way-10-shot and TACRED's 5-way-10-shot.

Method	T1	T2	T3	T4	T5	T6	T7	T8
ConPL	95.72	**93.53**	**91.31**	**89.95**	**88.93**	**88.39**	**87.43**	**85.77**
w.o. PM	95.72	93.11	89.90	87.96	86.68	85.53	84.36	82.21
w.o. CL	**96.43**	93.49	90.58	88.71	88.37	87.59	86.24	84.25
w.o. $\mathcal{L}_{cc}$	95.72	93.52	91.21	89.92	88.91	88.31	87.32	85.69
w.o. $\mathcal{L}_{dc}$	95.72	93.50	91.28	89.92	88.89	88.34	87.26	85.40
w.o. $\mathcal{L}_{fc}$	94.87	89.03	85.32	82.35	81.45	80.19	78.53	75.11

Table 2: Ablations experiments of FewRel's 10-way-5-shot.

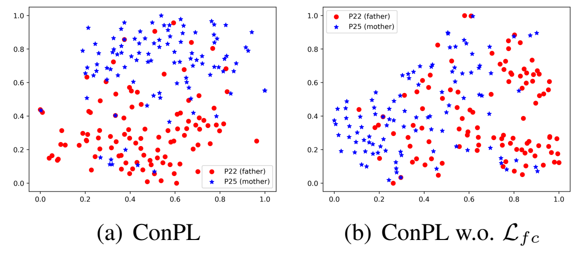


Figure 3: The influence of the focal loss $\mathcal{L}_{fc}$ on the embedding distribution of two similar classes ("father" and "mother") of validation set samples.

2-shot, FewRel's 10-way-10-shot, and TACRED's 5-way-10-shot. These results also show the effectiveness of our ConPL.

5.6 Ablation Study

We conduct ablation experiments on the FewRel's 10-way-5-shot from two aspects of the utilization of memory and auxiliary tasks to verify the contribution of each component in our method and study the impact on model performance by removing one module at a time. **Utilization of Memory** includes (a) without the Prototype Memory (PM), (b) without the Consistent Learning Module (CL), and **Auxiliary Tasks** includes (c) only removing the classification consistency loss $\mathcal{L}_{cc}$, (d) only removing the distribution consistency loss $\mathcal{L}_{dc}$, and (e) only removing the focal loss $\mathcal{L}_{fc}$. The results of our ablation experiments are shown in Table 2. We observe that each component in our method contributes to the overall performance.

In Utilization of Memory, Prototype Memory (PM) brings a 3.56% boost after training on all tasks, demonstrating the strong effectiveness of our novel multi-information episodic memory. The Consistent Learning Module (CL) gains a 1.52% improvement. Also, we find that training without CL requires fewer epochs than EMAR, ERDA, and RP-CRE, but outperforms these with PT by 2.91%, 7.23%, and 3.38%. The performance of T1 without CL is 0.71% higher than ConPL, as there is no catastrophic forgetting in the first task, causing the model to overfit on memory. Memory augmentation in T1 could reduce the diversity of the training samples, which leads to lower performance.

In Auxiliary Tasks, The classification consistency loss $\mathcal{L}_{cc}$ and the distribution consistency loss $\mathcal{L}_{dc}$ have a relatively small impact, with only 0.08% and 0.37% improvement, respectively. However, when $\mathcal{L}_{ce}$ and $\mathcal{L}_{fc}$ use sample memory to calculate probabilities, they can bring significant improvement. We will elaborate on this phenomenon in 5.7. In addition, we observe that the focal loss $\mathcal{L}_{fc}$ brings the most significant 10.66% improvement to the model's performance, demonstrating the importance of effectively dealing with the confusion of similar classes. Moreover, Figure 3 displays the t-SNE visualization results of the high similarity relations "father" and "mother", which show that our focal loss significantly improves the ability of the model to distinguish similar classes in NK-CRE.

5.7 Importance of Consistency Loss

To demonstrate the importance of the classification consistency loss and the distribution consistency loss, we conduct an analysis experiment on FewRel's 10-way-5-shot. Specifically, we replace the use of prototype memory with sample memory

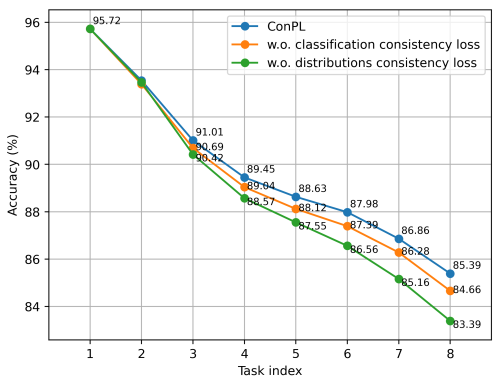


Figure 4: The analysis results of the importance of consistency loss.

to calculate probabilities in $\mathcal{L}_{ce}$ and $\mathcal{L}_{fc}$, which is a common prototype calculation method. In this case, we re-selected better hyperparameters and adjusted the relative weights λ_{fc} and λ_{dc} to 2 and 5, respectively. The ablation experiment is re-performed on (d) and (e) in 5.6, and the results are shown in Figure 4. The classification consistency loss and the distribution consistency loss improve by 0.73% and 2.0% respectively after T8. We think that due to the probabilities of $\mathcal{L}_{ce}$ and $\mathcal{L}_{fc}$ being computed by prototype memory, the model can learn correct classifications and inter-class discrepancy, leading to a relatively small improvement from the two consistency losses. However, when using sample memory to calculate probabilities, these two consistency losses can result in significant gains.

5.8 Forgetting

To provide a more intuitive comparison when testing forgetting, we added two baselines as lower and upper bounds.

SeqRun serves as a **lower bound**, where the model is directly fine-tuned on the new task data without using past data, leading to severe catastrophic forgetting.

JointTrain serves as an **upper bound**, where all training data for each new task, including previously seen data, is stored in the memory.

The forgetting of each task and their average forgetting on FewRel benchmark's 10-way-5-shot are presented in Table 3, along with the average forgetting on Fewrel's 10-way-2-shot, and TACRED's 5-way-5-shot in Figure 5. These results show that:

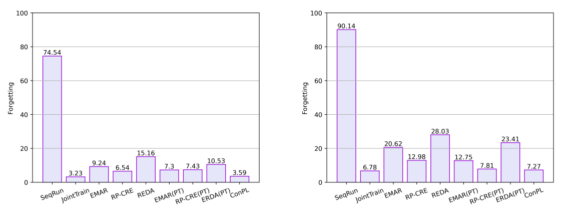
(1) The use of Prompt learning can significantly mitigate the issue of catastrophic forgetting.

(2) Compared with existing methods, our method greatly mitigates catastrophic forgetting in continual learning, which is crucial.

(3) Observing JointTrain, we find that forgetting still exists even when all the training data is available, and our continual learning method is the closest to it among all methods.

Method	T1	T2	T3	T4	T5	T6	T7	Mean
SeqRun	82.39	85.7	81.06	83.48	77.93	79.58	84.45	82.08
JointTrain	9.81	5.58	5.47	2.88	0.76	-0.22	-1.28	3.29
EMAR	23.39	21.76	17.16	13.95	6.38	7.27	3.48	13.34
RP-CRE	28.79	18.88	12.28	8.53	5.69	2.29	-1.58	10.7
ERDA	28.12	23.83	22.63	22.3	15.63	14.95	14.78	20.32
EMAR(PT)	11.83	9.22	8.64	8.4	6.17	3.74	2.73	7.24
RP-CRE(PT)	14.35	9.88	6.77	4.18	1.47	1.61	-1.98	5.18
ERDA(PT)	21.52	15.54	13.18	13.14	9.69	9.46	6.93	12.78
ConPL	**11.47**	**6.11**	**3.79**	**2.47**	**0.46**	**-0.12**	**-0.98**	**3.31**

Table 3: Forgetting (%) of various methods after training for each task on FewRel's 10-way-5-shot. "Mean" represents the average forgetting from T1 to T7.



(a) FewRel 10-way-2-shot (b) TACRED 5-way-5shot

Figure 5: The average forgetting (%) of various methods in different benchmarks.

6 Conclusion

We introduce **N**-way-**K**-shot **C**ontinual **R**elation **E**xtraction (NK-CRE), a novel problem in the realm of continual relation extraction, where all tasks are conducted in the N-way K-shot setting scenario. This problem presents a significant challenge, yet is highly applicable in real-world scenarios. To address this problem, we propose **Con**sistent **P**rototype **L**earning (ConPL), a novel method that effectively alleviates catastrophic forgetting caused by insufficient activation of old relations and the confusion among similar classes. Through a series of experimental and analytical results, we demonstrate that ConPL outperforms existing methods in NK-CRE. In future research, we plan to study how to improve the domain adaptability of the model in few-shot continual relation extraction.

Limitations

There are two limitations in this paper: (1) Compared with existing memory-based methods, the proposed prototype memory may bring additional storage space overhead. But since we only require very little additional memory (only one vector per class), we did not discuss it; (2) Although we noted that the distortion and forgetting of the prototype are highly correlated, we did not conduct a detailed analysis of the reasons for special prototypes that do not follow this pattern.

Acknowledgements

We would like to express our gratitude to the anonymous reviewers for their valuable feedback and suggestions, which have significantly improved the quality of our work. This research is supported by the Key Support Project of NSFC-Liaoning Joint Foundation (No. U1908216) and the Major Scientific Research Project of the State Language Commission in the 13th Five-Year Plan (No. WT135-38).

References

Sagie Benaim and Lior Wolf. 2018. One-shot unsupervised cross domain translation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. 2021. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9516–9525.

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

Kuilin Chen and Chi-Guhn Lee. 2021. Incremental few-shot learning via vector quantization in deep embedded space. In *International Conference on Learning Representations*.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web Conference 2022*, page 2778–2788.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 232–243.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

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.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Jiale Han, Bo Cheng, and Wei Lu. 2021a. Exploring task difficulty for few-shot relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2605–2616.

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, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021b. Ptr: Prompt tuning with rules for text classification.

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.

Chengwei Hu, Deqing Yang, Haoliang Jin, Zhen Chen, and Yanghua Xiao. 2022. Improving continual relation extraction through prototypical contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1885–1895.

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.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Abiola Obamuyide and Andreas Vlachos. 2019. Meta-learning improves lifelong relation extraction. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 224–229.

Chengwei Qin and Shafiq Joty. 2022. Continual few-shot relation learning via embedding space regularization and data augmentation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2776–2789.

Meng Qu, Tianyu Gao, Louis-Pascal Xhonneux, and Jian Tang. 2020. Few-shot relation extraction via Bayesian meta-learning on relation graphs. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 7867–7876.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5569–5578.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Yao-Hung Hubert Tsai and Ruslan Salakhutdinov. 2017. Improving one-shot learning through fusing side information.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059.

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, Volume 1 (Long and Short Papers)*, pages 796–806.

Peiyi Wang, Yifan Song, Tianyu Liu, Binghuai Lin, Yunbo Cao, Sujian Li, and Zhifang Sui. 2022. Learning robust representations for continual relation extraction via adversarial class augmentation.

Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. 2020. Supermasks in superposition. In *Advances in Neural Information Processing Systems*, volume 33, pages 15173–15184. Curran Associates, Inc.

Kaijia Yang, Nantao Zheng, Xinyu Dai, Liang He, Shujian Huang, and Jiajun Chen. 2020. Enhance prototypical network with text descriptions for few-shot relation classification. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management*, page 2273–2276.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3987–3995. JMLR.org.

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.

Kang Zhao, Hua Xu, Jiangong Yang, and Kai Gao. 2022. Consistent representation learning for continual relation extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3402–3411.

A Appendix

A.1 Prototype Distortion and Forgetting

To evaluate the relationship between the change of feature embeddings and the degree of forgetting, we introduce distortion and forgetting metrics for each class prototype. Distortion is calculated using the following formula:

$$D_r = \frac{1}{n-i}\sum_{j=i+1}^{n} d_r^j \tag{13}$$

$$d_r^j = 1 - s(e_{i,r}, e_{j,r}) \tag{14}$$

where i of $e_{i,r}$ means that the relation r firstly appears in task i and $e_{i,r}$ denotes the embedding of the relation r after task i is trained. s represents the calculation formula of cosine similarity. D_r refers to the mean value of the distortion for relation r from task i to task n.

Method	T1	T2	T3	T4	T5	T6	T7	T8	Mean
EMR	$96.35_{\pm 0.25}$	$88.02_{\pm 2.09}$	$78.83_{\pm 2.80}$	$75.15_{\pm 2.85}$	$72.00_{\pm 2.23}$	$69.41_{\pm 2.06}$	$66.70_{\pm 1.57}$	$63.68_{\pm 1.47}$	$76.27_{\pm 1.92}$
EMAR	$92.03_{\pm 1.98}$	$78.87_{\pm 3.72}$	$72.81_{\pm 5.25}$	$69.19_{\pm 4.45}$	$68.05_{\pm 4.08}$	$66.23_{\pm 1.95}$	$63.68_{\pm 2.55}$	$61.77_{\pm 1.48}$	$71.58_{\pm 3.18}$
IDLVQ-C	$96.03_{\pm 0.12}$	$87.18_{\pm 2.51}$	$76.63_{\pm 3.97}$	$73.57_{\pm 4.43}$	$67.74_{\pm 3.60}$	$65.16_{\pm 2.96}$	$62.64_{\pm 1.87}$	$60.32_{\pm 1.87}$	$73.66_{\pm 2.65}$
ERDA	$96.38_{\pm 0.35}$	$88.91_{\pm 1.96}$	$83.10_{\pm 1.80}$	$79.73_{\pm 2.69}$	$74.83_{\pm 3.06}$	$72.84_{\pm 1.75}$	$70.28_{\pm 1.79}$	$68.07_{\pm 1.94}$	$79.26_{\pm 1.92}$
EMAR†	$82.50_{\pm 10.86}$	$76.23_{\pm 17.56}$	$71.41_{\pm 10.67}$	$66.93_{\pm 14.52}$	$62.02_{\pm 16.71}$	$62.37_{\pm 8.23}$	$60.05_{\pm 6.30}$	$59.29_{\pm 2.78}$	$67.59_{\pm 10.95}$
RP-CRE†	$86.38_{\pm 28.53}$	$78.47_{\pm 28.71}$	$71.93_{\pm 24.10}$	$69.07_{\pm 5.69}$	$66.28_{\pm 11.46}$	$65.88_{\pm 2.52}$	$62.28_{\pm 6.64}$	$61.17_{\pm 1.08}$	$70.18_{\pm 13.59}$
ERDA†	$88.15_{\pm 17.17}$	$79.65_{\pm 13.60}$	$74.05_{\pm 11.92}$	$68.18_{\pm 13.43}$	$67.14_{\pm 8.51}$	$66.96_{\pm 4.61}$	$65.86_{\pm 3.44}$	$63.60_{\pm 5.17}$	$71.70_{\pm 9.73}$
EMAR(PT)†	$95.28_{\pm 5.17}$	$92.75_{\pm 4.42}$	$90.57_{\pm 1.97}$	$88.15_{\pm 2.99}$	$87.04_{\pm 1.46}$	$84.90_{\pm 1.83}$	$83.07_{\pm 2.47}$	$81.34_{\pm 1.97}$	$87.89_{\pm 2.76}$
RP-CRE(PT)†	$93.52_{\pm 3.44}$	$89.48_{\pm 9.21}$	$87.67_{\pm 8.55}$	$85.28_{\pm 2.98}$	$84.48_{\pm 2.43}$	$83.08_{\pm 2.02}$	$81.92_{\pm 1.44}$	$80.87_{\pm 0.89}$	$85.79_{\pm 1.92}$
ERDA(PT)†	$\mathbf{96.55}_{\pm 3.36}$	$92.56_{\pm 4.98}$	$88.56_{\pm 4.14}$	$84.47_{\pm 12.77}$	$84.14_{\pm 3.10}$	$79.94_{\pm 4.26}$	$78.45_{\pm 2.99}$	$77.02_{\pm 6.59}$	$85.21_{\pm 5.27}$
ConPL	$95.72_{\pm 4.27}$	$\mathbf{93.53}_{\pm 2.30}$	$\mathbf{91.31}_{\pm 2.93}$	$\mathbf{89.95}_{\pm 1.02}$	$\mathbf{88.93}_{\pm 0.50}$	$\mathbf{88.39}_{\pm 0.26}$	$\mathbf{87.43}_{\pm 0.13}$	$\mathbf{85.77}_{\pm 0.56}$	$\mathbf{90.12}_{\pm 1.50}$

Table 4: Whole accuracy (%) and variance of different methods after training for each task on **FewRel** benchmark in the **10-way-5-shot** (NK-CRE) setting. † denotes our reproduced results with the publicly available codebases, where PT represents using prompt learning to enrich the inputs. Other results are obtained directly from Qin and Joty (2022) in the CFRL setting. The best values are denoted in **bold**. The "Mean" column shows the mean of the whole accuracy and variance of T1∼T8.

The formula for the forgetting of the prototype is as follows:

$$F_r = \frac{1}{n-i} \sum_{j=i+1}^{n} g_r^j \tag{15}$$

$$g_r^j = \max_{l \in \{i, \cdots, j-1\}} a_{l,r} - a_{j,r} \tag{16}$$

where $a_{l,r}$ represents the whole accuracy of the test set for relation r after the task l is trained, f_r^j represents the degree of forgetting of relation r after task j is trained, and i represents the relation r firstly appears in task i.

Based on the above formulas, we calculate the distortion and forgetting of each prototype in different methods. Considering that the order of task sequences has a relatively large impact on the change of embedding and forgetting of each class, we randomly generate 50 task sequences and show their average results. Figure 1 illustrates the correlation between the embedding distortion and the degree of forgetting in different methods.

A.2 Datasets

The following is the detailed introduction of the FewRel (Han et al., 2018) and TACRED (Zhang et al., 2017) datasets.

FewRel is a few-shot relation classification dataset that was proposed by Han et al. (2018). It consists of 100 relation classes, each of which has 700 instances. The dataset is split into a training set, a validation set, and a test set, with a split of 64/16/20 fraction corresponding to each set. Following CFRL (Qin and Joty, 2022), we randomly divide the publicly accessible 80 relations from the training and validation sets into 8 tasks, each containing 10 relations (10-way). In this paper, we set 2-shot, 5-shot, and 10-shot as different few-shot settings in the NK-CRE benchmark.

TACRED is a relation extraction dataset that was proposed by Zhang et al. (2017). It contains 42 relations and over 100,000 instances. The dataset is preprocessed by filtering out the special relation "n/a", resulting in the remaining 41 relation classes and 68,438 instances to construct the NK-CRE benchmark. Similar to **FewRel**, we divide the 41 relations into 8 tasks, with one task containing 6 relations and the others containing 5 relations (5-way). We set 5-shot and 10-shot as different few-shot settings in this paper.

A.3 Main Results with Means and Variances

Table 4 shows the whole accuracy (%) and variance of different methods after training for each task on FewRel's 10-way-5-shot. We observe that due to the strict implementation of the few-shot continual learning task, NK-CRE has a larger variance compared to CFRL. Nonetheless, the average variance of our method ConPL is still lower than that of existing methods in CFRL, indicating that our method is robust.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations

☑ A2. Did you discuss any potential risks of your work?
Limitations

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract; 1 Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5 Experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
5.4 Training Details

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5.4 Training Details

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5.5 Main Results; 5.6 Ablation Study; 5.7 Importance of Consistency Loss; 5.8 Forgetting

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5.4 Training Details

D ☒ **Did you use human annotators (e.g., crowdworkers) or research with human participants?**
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.