Continual Few-shot Relation Extraction via Adaptive Gradient Correction and Knowledge Decomposition

Anonymous ACL submission

Abstract

Continual few-shot relation extraction (CFRE) aims to continually learn new relations with limited samples. However, current methods neglect the instability of embeddings in the process of different task training, which leads to serious catastrophic forgetting. In this paper, we propose the concept of the following degree from the perspective of instability to analyze catastrophic forgetting and design a novel method based on adaptive gradient correction and knowledge decomposition to alleviate catastrophic forgetting. Specifically, the adaptive gradient correction algorithm is designed to limit the instability of embeddings, which adaptively constrains the current gradient to be orthogonal to the embedding space learned from previous tasks. To reduce the instability between samples and prototypes, the knowledge decomposition module decomposes knowledge into general and task-related knowledge from the perspective of model architecture, which is asynchronously optimized during training. Experimental results on two standard benchmarks show that our method outperforms the state-of-the-art CFRE model and effectively improves the following degree of embeddings.

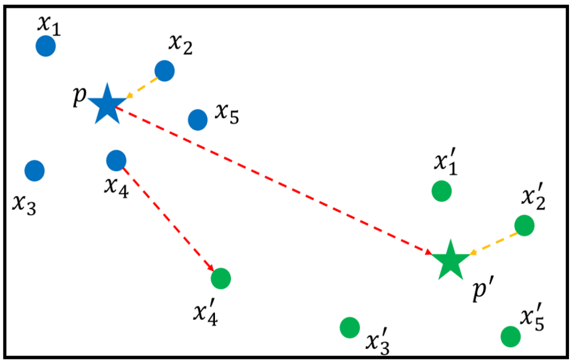

Figure 1: Representation of distance relative offset distance and absolute offset distance. $\{x_1, \cdots, x_5\}$ and $\{x'_1, \cdots, x'_5\}$ are sample embeddings with the same class before and after training at a certain time step, respectively. p and p' are embeddings of prototypes of those. The red dashed arrow represents the AOD of the sample x_4 or prototype p. The difference distance of the two yellow dotted arrows represents ROD between x_2 and p.

1 Introduction

The purpose of CFRE is to continuously train a model on limited new data. Compared to traditional continual relation extraction(CRE) models, it can learn new relations without accessing a large number of previous task data, and avoid catastrophic forgetting (French, 1999; McCloskey and Cohen, 1989) of the old relations.

Due to limited training data, the features learned by the model at each time step of continual learning are relatively unstable and are easily modified by the model when learning other class samples in subsequent time steps. As a result, traditional CRE methods cannot be directly applied to CFRE (Qin and Joty, 2022). To fully utilize data resources, scholars have explored many methods (Wang et al., 2022b; Zhong et al., 2021; Zhang et al., 2022). And the methods based on memory replay (Chen et al., 2023; Wang et al., 2023; Qin and Joty, 2022) made great achievements in CFRE. However, these models mainly focus on the strategies of memory samples in the process of replaying or learning and perform direct fine-tuning of the model parameters through the gradient of the loss function. Furthermore, in-memory examples are used to generate gradients that benefit the performance on previous tasks, but the direction optimized by these gradients may contradict the optimization direction of the current task gradient, which leads to the instability of samples with respect to prototypes. In metric learning (Kaya and Bilge, 2019), the prototype of one class is the center of this class.

In essence, during continuous learning with minimal data usage, when samples are equivalently shifted with the embedding of prototypes, the labels of samples will not be easily changed and catastrophic forgetting of the model will be alleviated. Thus, we formally provide the definition of **following degree** of class y at time step t: the degree of deviation between class y samples and class y prototype before and after training at t. Obviously, the smaller the offset between the previous sample embedding and its prototype, the higher the following degree of this class, and the less unstable the samples are. In this way, the model can have less catastrophic forgetting of previous samples.

Intuitively, there are two main reasons for the low following degree of the model on a task: 1) the model is not sufficiently optimized for some samples, which makes the prototypes deviate from the optimization direction of the samples; 2) after optimization, all embeddings change too much compared with the previous will also increase the risk of low following degree. **Relative offset distance (ROD)** and **absolute offset distance (AOD)** are proposed to express the above two factors, respectively. As shown in Fig. 1, $|x_4'-x_4|$ and $|p'-p|$ represent AOD of x_4 and p. $(|p' - x_2'| - |p - x_2|)$ represents ROD between x_2 and p. The lower the AOD and ROD, the higher the following degree.

In this paper, an adaptive gradient correction algorithm is proposed to directly constrain and correct the vector space of the gradient in transformer-based language models. Since the modified gradients are orthogonal to the embeddings of the previous sample, this special optimization method for the transformer can effectively reduce AOD. Specifically, for each previous task, we calculate a gradient direction, which has the greatest impact on in-memory samples, as the correction criteria. When updating the gradient of the current task, the correction matrix based on this criteria is used to make a linear transformation of the gradient, which can constrain the subspace of parameters orthogonal to the previous tasks.

In addition, knowledge contained in model parameters is decomposed into general and task-related knowledge. Based on this decomposition, general knowledge is used to identify a generic representation of relations with the corresponding entities. We employ task-related knowledge to identify categorical decision boundaries between specific tasks based on general knowledge. In practice, we apply a pre-trained language model (PLM) to encode the general knowledge and use an adaptive gradient correction algorithm to avoid mutual coverage of knowledge. Since there is no gradient transmission between task prototypes, we integrate task-related knowledge into task prototype embedding. These prototypes are updated discretely and continuously in three different training stages to reduce ROD. In the continuous optimization stage, we add an additional loss to increase the distance between prototypes, which can prevent confusion between the new and the old task prototypes.

To sum up, the contributions of this paper mainly include the following three aspects:

- We attribute catastrophic forgetting in CFRE to the low following degree between samples and prototypes, and analyze how to improve the following degree from the perspectives of AOD and ROD.
- According to the correction matrix calculated by in-memory samples, an adaptive gradient correction algorithm is proposed that makes the model directly adjust the gradient to reduce AOD.
- We design a knowledge decomposed method and corresponding update strategies to avoid the confusion of knowledge between different tasks, which can limit ROD during training.

Experimental results on two public datasets show that our method can effectively alleviate the catastrophic forgetting in CFRE.

2 Related Work

RE aims to extract the implied relation from sentences. For example, given the sentence "Steve Jobs is the co-founder of Apple", the model needs to determine the relation "CEO_of" between the entity "Steve Jobs" and "Apple". It is a basic step for many downstream tasks such as language understanding, question answering, and knowledge graph construction (Nasar et al., 2021).

Most traditional RE models are built based on a fixed dataset (Eberts and Ulges, 2020; Liu et al., 2022b; Shang et al., 2022; Li et al., 2018, 2021). However, RE is often an open vocabulary problem (Liu et al., 2022a), and it is difficult to model all relations for any limited set. Therefore, the continual learning ability (Chen and Liu, 2018) of the RE model has gradually attracted attention (Zhao et al., 2023; Xia et al., 2023).

But there will be serious catastrophic forgetting for the model in continual learning, that is, the model will forget the old knowledge when learning new tasks. To solve this problem, the existing methods for continual learning mainly include: regularization-based approach (Zhai et al., 2019), memory-based approach (Cha et al., 2021), optimization-based approach (Schwarz et al., 2018), representation-based approach (Yan et al., 2022), and architecture-based approach (Wang et al., 2022a). The existing methods (Wang et al., 2022c; Zhou et al., 2022) are often based on a large number of labeled data for training, which is time-consuming and expensive. When the number of training is small, overfitting of memory data and knowledge coverage (Song et al., 2023) is also one of the causes of catastrophic forgetting. The existing methods mainly solve this problem from the following three levels: data level (Wang et al., 2022b), feature level (Zhong et al., 2021), and task level (Zhang et al., 2022).

Directly adjusting the gradient is also one of the methods to overcome catastrophic forgetting with limited data (Chen et al., 2023; Zhou et al., 2022). He and Jaeger (2018) proposed Concept Aided BackProp for disaster forgetting, in which the gradient is shielded by a conceptor to prevent the degradation of previous tasks. Zeng et al. (2019) introduced orthogonal projection and context-dependent processing module for the current gradient. Guo et al. (2022) put forward the paranoid factor related to the previous task to estimate the input space, but the accumulation strategy for embedding of different samples may lead to confusion in the projection space. Although, the GEM algorithm (Lopez-Paz and Ranzato, 2017) introduced gradient projection to make the loss of previous tasks slowly increase in subsequent training, the embeddings computed by this method are not equivariant and cannot be applied to transformer-based language models.

In contrast, our method innovatively uses the correction matrix calculated by in-memory samples to directly constrain the gradient update in backpropagation to reduce the overall AOD, so as to limit the risk of reduced following degree. A model architecture based on knowledge decomposition and the corresponding asynchronous optimization method is designed to further reduce the ROD of samples.

3 Methodology

3.1 Problem Definition

CFRE model extracts the relations from a series of tasks $\{T^1, T^2, \cdots, T^K\}$. Every task T^k has its own training dataset D^k_{train}, validation dataset D^k_{valid}, test dataset D^k_{test} and relation label set R^k. Each set contains a small number of sample pairs $\{(x_i, y_i)\}_{i=1}^{|D|}$, where the label $y_i \in R^k$. For example, in N-way M-shot constraint, we make $|R^k| = N$ and $|D^k_{train}| = N \times M$. At time step k, the model will only train on D^k_{train} and we hope that the model will perform well on $\{D^1_{test} \cup D^2_{test} \cup \cdots \cup D^k_{test}\}$ after training.

The memory mechanism is applied to prevent catastrophic forgetting in CFRE. Memory is defined as a series of sample sets $\hat{M}^K = \cup_{j=1}^{K} M^j$, where each $M^k = \{(x_i, y_i)\}_{i=1}^{|M^k|}$ corresponds to a task T^k. At time step k, a portion of the sample is selected to be stored in the memory set M^k as classical samples. In this paper, only one sample is in memory for each category.

3.2 Training Process

Algorithm. 1 describes the whole training process of our method at time step k. It mainly includes three different training stages to continuously and discretely adjust ROD between prototypes and samples.

In the first stage, we initialize the current memory set M^k and proto-embedding P^k. Then, the temporary memory $\{\hat{M}^{k-1} \cup M^k\}$ with all training data is applied to preliminarily update the PLM parameters θ^{k-1} through the corrected gradient.

In the second stage, we introduce cosine similarity to find one sample that is most similar to the center of the category cluster and add it to memory $\hat{M}^{k-1}$. The PLM parameters update is similar to the first stage, except for using selected memory data.

In the final stage, we freeze the gradient of PLM and view proto-embeddings $\hat{P}^k$ as learnable parameters, fine-tuning them on training data.

The data and gradient flow are shown in Fig. 2.

3.3 Knowledge Decomposition

To prevent the parameters in the model from being covered and confused due to continuous task iteration, we abstractly decompose them into general and task-related knowledge. Since they do not interfere with each other during the update process, ROD for different categories can be effectively reduced.

3.3.1 General knowledge encoder

Because of the training with mask prediction on a large number of corpora, the knowledge of BERT parameters is independent of specific downstream

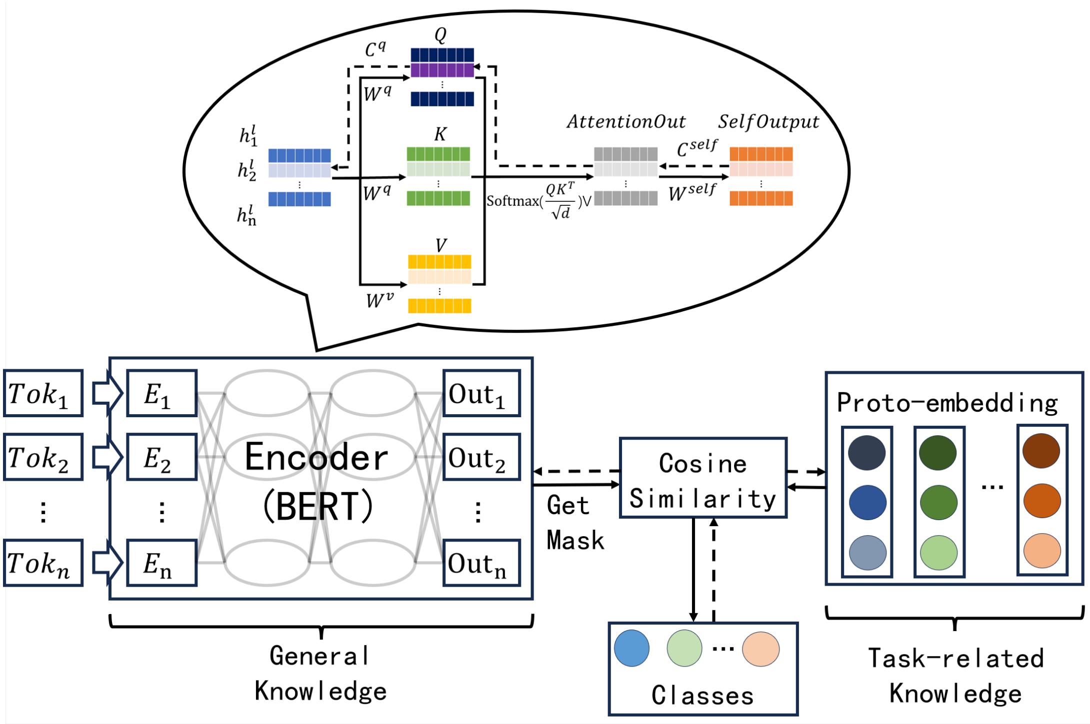

Figure 2: The data and gradient flow of our method, where the solid lines describe the data flow in forward propagation and the dashed lines describe the gradient flow in backpropagation.

tasks. Therefore, we choose BERT as PLM to encode the general knowledge of the whole relation extraction model. Specifically, for a sentence $x = \{Tok_1, Tok_2, \cdots, Tok_n\}$, we first construct a template function based on prompt learning (Liu et al., 2023) by adding a special $[MASK]$ token to the sentence and get the sequence $x_{seq} = \{[CLS], e_h, [MASK], e_t, [SEP], x, [SEP]\}$. e_h and e_t respectively represent the head and tail entity in the sentence x. Then the embedding of $[MASK]$ token is taken as the relation representation of the whole sentence:

$$h_{[MASK]} = f_\theta(x_{seq}) \quad (1)$$

To facilitate symbolic representation, x_{seq} is replaced by x.

3.3.2 Task-related knowledge encoder

To ensure that the new task knowledge learned by the model does not conflict with the previous task and model general knowledge respectively, we encode the knowledge of different tasks into the corresponding category proto-embedding. Since the proto-embeddings of different categories are disconnected, the knowledge from new and previous tasks will not affect each other naturally. And this part of knowledge is updated and trained asynchronously with BERT, so it also does not conflict with the general knowledge. The proto-embeddings are updated with different strategies in the three stages of training.

In the first stage of task k, we apply all samples of each new relation in the current task to calculate the new proto-embedding. To get the most representative vector in the sample embedding space, the average method is introduced to aggregate all embeddings of samples. For the proto-embedding p_j^k of class j task k, the calculation formula is as follows:

$$P_j^k = \frac{1}{|D_j^k|} \sum_{(x_i, y_i) \in D_j^k} f_\theta(x_i) \quad (2)$$

where $D_j^k = \{(x_i, y_i) | (x_i, y_i) \in D_{train}^k, y_i = r_j\}$ and $|D_j^k|$ is the number of samples in D_j^k.

In the second stage, there is only one sample for each class in memory, so we directly use the relation embedding of that sample as the proto-embedding of the corresponding class.

The updating of proto-embeddings in the first two stages is discrete. In the third stage, we freeze the parameters of BERT and regard the proto-embedding as a parameter that can be updated by the gradient descent algorithm. The proto-embeddings are continuously updated by the gradi-

Algorithm 1 Training procedure for $T^k (k > 1)$

Input: The PLM parameters θ^{k-1} and proto-embedding $\hat{P}^{k-1} = \cup_{j=1}^{k-1} P^j$ trained on T^{k-1}, training data set D_{train}^k, memory set $\hat{M}^{k-1}$, learning rate γ

Output: θ^k, P^k

Use all samples in D_{train}^k to initialize M^k and P^k
Freeze the gradient of $\hat{P}^{k-1}$, unfreeze the gradient of θ^{k-1}
for $i \in \{1, \cdots, epoch_1\}$ **do**
  Calculate correction matrix C using $\hat{M}^{k-1}$
  Calculate corrected gradient $\Delta\theta^{k-1}$ using C with L_{gen} on $D = \{\hat{M}^{k-1} \cup M^k \cup D_{train}^k\}$
  Update $\theta^{k-1} \leftarrow \theta^{k-1} - \gamma\Delta\theta^{k-1}$
  Recalculate P^k using M^k
end for
Select typical samples from D_{train}^k to update M^k
$\hat{M}^k \leftarrow \hat{M}^{k-1} \cup M^k$
for $i \in \{1, \cdots, epoch_2\}$ **do**
  Calculate correction matrix C using $\hat{M}^{k-1}$
  Calculate corrected gradient $\Delta\theta^{k-1}$ using C with L_{gen} on $D = \{\hat{M}^k \cup D_{train}^k\}$
  Update $\theta^{k-1} \leftarrow \theta^{k-1} - \gamma\Delta\theta^{k-1}$
  Recalculate P^k using M^k
end for
$\hat{P}^k \leftarrow \hat{P}^{k-1} \cup P^k$
Freeze the gradient of θ^k, unfreeze the gradient of $\hat{P}^k$
for $i \in \{1, \cdots, epoch_3\}$ **do**
  Calculate gradient $\Delta\hat{P}^k$ with L_{task} on $D = D_{train}^k$
  Update $\hat{P}^k \leftarrow \hat{P}^k - \gamma\Delta\hat{P}^k$
end for

ent.

$$p_j^k \leftarrow p_j^k - \gamma \frac{\partial L_{task}}{\partial p_j^k} \tag{3}$$

where γ is the learning rate. L_{task} is calculated using cross entropy function, which is similar to Eqn. 5. We set $P^k = \cup_{j=1}^{|R^k|} p_j^k$.

3.3.3 Calculation of loss function

To optimize the learned embeddings of relations, the training and inference of our model are based on metric learning. At time step k, by measuring the similarity between each sample x_i and proto-embedding $\hat{P}^k$, the relation distribution is calculated as:

$$\mathtt{p}(r_i|x_i) = \frac{exp(d(f_\theta(x_i), p_i))}{\sum_{l=1}^{|P^k|} exp(d(f_\theta(x_i), p_l))} \tag{4}$$

where $d(\cdot, \cdot)$ is the distance metric function (cosine similarity in this paper) and p_l is proto-embedding in $\hat{P}^k$.

Further, the cross entropy loss function is calculated to measure the classification error of the model:

$$L_{ce} = - \sum_{(x_i, y_i) \in D} log \mathtt{p}(r_i|x_i) \tag{5}$$

While alleviating catastrophic forgetting, it is also important to correctly handle the information entropy of new tasks in the model. To avoid the confusion of similar relation between new task and previous tasks, we select the proto-embedding sets $P_i^{sim} = \{p_l | d(f_\theta(x_i), p_i) - d(f_\theta(x_i), p_l) < \alpha\}$ and $P_i^{neg} = \{p_l | max(d(f_\theta(x_i), p_l), l \neq i\}$, which are easy to be confused with the correct category for each sample x_i. α is the set similarity threshold. The following probability is reduced by cross entropy loss function:

$$L_{sim} = - \sum_{(x_i, y_i) \in D} log \frac{exp(d(f_\theta(x_i), p_i))}{\sum_{l=1}^{|P_i^{sim} \cup P_i^{neg} \cup p_i|} exp(d(f_\theta(x_i), p_l))} \tag{6}$$

The final general knowledge loss is calculated as:

$$L_{gen} = L_{ce} + L_{sim} \tag{7}$$

3.4 Adaptive Gradient Correction

Through the decomposition of parameters, the task-related knowledge will not interfere with each other and ROD of previous tasks will be kept at a low level in the process of continual learning. However, when updating the parameters of the general knowledge encoder, there will still be gradient interference between tasks, which leads to the coverage of the general knowledge in BERT. These coverage may make the embedding and AOD of previous task sample change dramatically, which will increase the risk of reducing the following degree.

To avoid the interference of previous task embeddings when BERT learns a new task, the memory embedding of the corresponding hidden layer is extracted to adaptively correct the feed forward networks (FFN) and query matrix gradient in each layer during the model back-propagation. And the other gradients in BERT are frozen besides these two kinds of gradients.

To express simplicity, we first abstract all layers of the encoder into fully connected layers to introduce our idea. At time step k, we first extract the hidden state $H^l \in R^{d\times|\hat{M}^{k-1}|}$ of samples in $\hat{M}^{k-1}$ before inputting each unfrozen layer. Then the gradient correction matrix C^l of a certain layer is calculated according to H^l (The superscripts are omitted for clarity):

$$C = I - H(H^T H)^{-1}H^T \tag{8}$$

Obviously, we can get $CH = H^T C^T = 0$. To guarantee the reversibility in the specific calculation, we add a small offset: $(\alpha I + H^T H)^{-1}$. The calculation method of α is consistent with Guo et al. (2022).

When using the back-propagation algorithm to calculate the gradient of a certain layer, the unfrozen gradient of BERT is corrected by C^l. The parameter W^l update formula for a certain layer is shown as follows:

$$\begin{aligned} \Delta W^l &= \frac{\partial L_{gen}}{\partial W^l} C^l \\ W^l &\leftarrow W^l - \gamma \Delta W^l \end{aligned} \tag{9}$$

where γ is the learning rate.

For the parameters in a transformer, we only update the FFN layer parameters and the parameters related to the query matrix in the self-attention layer. The method of updating parameters in the FFN layer is similar to Equation 9. The gradient update method of the query matrix in the self-attention layer is designed as,

$$\begin{aligned} \Delta W^q &= (C^q)^T \frac{\partial L_{gen}}{\partial W^q} \\ W^q &\leftarrow W^q - \gamma \Delta W^q \end{aligned} \tag{10}$$

where $(C^q)^T$ is the transpose row of matrix C^q, W^q is parameters to calculate query matrix. The detailed discussion and further proof are included in the Appendix A.

4 Experiments

4.1 Experimental Setup

Datasets Consistent with previous work on CFRE (Chen et al., 2023), our experiment will be conducted on two common datasets. FewRel (He and Jaeger, 2018) is a RE dataset that includes 80 relations, with 700 samples of each relation. We divide these relations into 8 tasks $\{T^1, \cdots, T^8\}$, where each task contains 10 relations. M samples are randomly drawn to form D^k_{train} with the constraint of 10-way M-shot. TACRED (Zhang et al., 2017) is a large-scale RE dataset based on news networks and online documents, containing 42 relation labels and 106,264 samples. Samples in TACRED are imbalanced compared with FewRel. We remove the particular relation "n/a" (not available) and divide the remaining 41 relations into eight subsets. The first subset has one more relation than other subsets

Evaluation Metrics At time step k, we test the model on $\hat{D}^k_{test}$, the union of all visible relation test sets, which can simultaneously reflect the model performance on new and old tasks. Since CFRE may be affected by the task sequence, we run random seeds six times on different task sequences to ensure the randomization of that. The mean and variance of relation classification accuracy on six different task sequences are introduced as the performance of the model. The training details are discussed in Appendix B.

Baselines Four baselines are introduced to compare our method (AGCKD). **CEAR** (Zhao et al., 2023) is a CRE approach that designs memory-insensitive relation prototypes and memory augmentation to overcome the overfitting problem. **SCKD** (Wang et al., 2023) is a contrastive learning scheme for CFRE, which employs serial knowledge distillation and pseudo-samples for contrastive learning to keep the representation of samples in different relations distinguishable. In **ERDA** (Qin and Joty, 2022), the embedding spatial regularization and data augmentation algorithms are proposed to optimize memory expression in CFRE tasks. **ConPL** (Chen et al., 2023) is the state-of-the-art method for CFRE. Prototype-based classification module, memory enhancement module, and consistency learning module are used to enhance the consistency of distribution as well as avoid catastrophic forgetting. In addition to these baselines, we also added two experimental settings to observe the upper and lower limits. In **Joint Training** setting, the model saves all training samples in the memory. Since the model can replay all past data at every time step, there is no catastrophic forgetting. In **SeqRun** setting, the memory does not save any samples. This setting may cause the

Method	Task index							
	1	2	3	4	5	6	7	8
				10-way 5-shot of FewRel				
Joint Training	$97.93_{\pm0.47}$	$96.25_{\pm0.35}$	$94.09_{\pm0.45}$	$92.37_{\pm0.39}$	$91.96_{\pm0.71}$	$91.15_{\pm0.56}$	$90.35_{\pm0.40}$	$88.9_{\pm0.03}$
SeqRun	$97.06_{\pm1.34}$	$92.77_{\pm1.36}$	$85.76_{\pm3.06}$	$80.95_{\pm2.82}$	$75.19_{\pm3.87}$	$66.62_{\pm3.71}$	$55.53_{\pm2.05}$	$42.57_{\pm2.26}$
CEAR	$69.46_{\pm7.49}$	$64.53_{\pm1.7}$	$62.22_{\pm3.01}$	$61.27_{\pm3.88}$	$60.04_{\pm2.37}$	$58.70_{\pm3.51}$	$57.88_{\pm2.66}$	$55.77_{\pm2.63}$
SCKD	$94.77_{\pm0.35}$	$82.83_{\pm2.61}$	$76.21_{\pm1.61}$	$72.19_{\pm1.33}$	$70.61_{\pm2.24}$	$67.15_{\pm1.96}$	$64.86_{\pm1.35}$	$62.98_{\pm0.88}$
ERDA	$96.55_{\pm0.43}$	$92.56_{\pm2.29}$	$88.56_{\pm3.34}$	$84.47_{\pm3.25}$	$84.14_{\pm3.01}$	$79.94_{\pm2.46}$	$78.45_{\pm1.74}$	$77.02_{\pm2.93}$
ConPL	$95.23_{\pm2.29}$	$92.77_{\pm2.78}$	$90.58_{\pm2.17}$	$89.03_{\pm0.96}$	$88.64_{\pm1.39}$	$88.09_{\pm1.06}$	$87.29_{\pm0.95}$	$85.83_{\pm0.62}$
AGCKD (Ours)	$\mathbf{97.78_{\pm0.95}}$	$\mathbf{95.93_{\pm1.30}}$	$\mathbf{94.13_{\pm0.81}}$	$\mathbf{92.67_{\pm0.40}}$	$\mathbf{91.71_{\pm0.79}}$	$\mathbf{90.97_{\pm0.65}}$	$\mathbf{90.11_{\pm0.53}}$	$\mathbf{88.68_{\pm0.47}}$
				5-way 5-shot of TACRED				
Joint Training	$97.93_{\pm0.47}$	$96.25_{\pm0.35}$	$94.09_{\pm0.45}$	$92.37_{\pm0.39}$	$91.96_{\pm0.71}$	$91.15_{\pm0.56}$	$90.35_{\pm0.40}$	$88.9_{\pm0.03}$
SeqRun	$97.06_{\pm1.34}$	$92.77_{\pm1.36}$	$85.76_{\pm3.06}$	$80.95_{\pm2.82}$	$75.19_{\pm3.87}$	$66.62_{\pm3.71}$	$55.53_{\pm2.05}$	$42.57_{\pm2.26}$
CEAR	$82.14_{\pm7.28}$	$68.43_{\pm8.46}$	$57.43_{\pm6.80}$	$51.83_{\pm6.75}$	$48.71_{\pm6.04}$	$45.23_{\pm4.25}$	$43.29_{\pm2.88}$	$40.74_{\pm4.08}$
SCKD	$88.42_{\pm0.83}$	$79.35_{\pm4.13}$	$70.61_{\pm3.16}$	$66.78_{\pm4.29}$	$60.47_{\pm3.05}$	$58.05_{\pm3.84}$	$54.41_{\pm3.47}$	$52.11_{\pm3.15}$
ERDA	$94.57_{\pm2.72}$	$86.55_{\pm3.55}$	$78.59_{\pm2.88}$	$74.58_{\pm3.92}$	$69.31_{\pm1.63}$	$66.53_{\pm3.12}$	$61.92_{\pm4.61}$	$55.97_{\pm2.16}$
ConPL	$96.79_{\pm3.01}$	$88.65_{\pm4.61}$	$85.40_{\pm4.66}$	$82.67_{\pm2.67}$	$80.82_{\pm2.79}$	$79.46_{\pm3.26}$	$77.47_{\pm2.34}$	$75.82_{\pm1.12}$
AGCKD (Ours)	$\mathbf{98.85_{\pm1.37}}$	$\mathbf{91.43_{\pm3.17}}$	$\mathbf{87.89_{\pm3.89}}$	$\mathbf{85.04_{\pm2.26}}$	$\mathbf{83.12_{\pm1.69}}$	$\mathbf{81.99_{\pm2.34}}$	$\mathbf{80.48_{\pm2.24}}$	$\mathbf{78.56_{\pm1.10}}$

Table 1: Accuracy (%) of various methods for each task on Fewrel's 10-way 5-shot and TACRED's 5-way 5-shot.

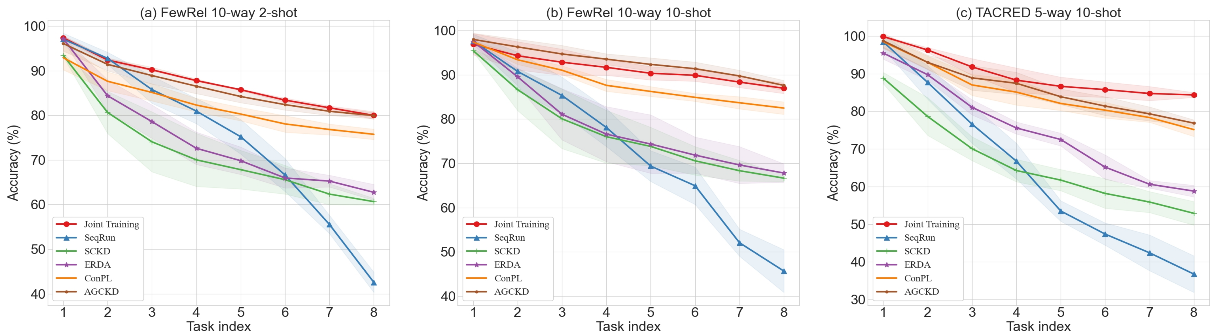

Figure 3: Comparison results for each task on Fewrel's 10-way 2-shot, Fewrel's 10-way 10-shot and TACRED's 5-way 10-shot. The variance is reported as light color regions.

model to face severe catastrophic forgetting, so it serves as a lower bound.

4.2 Main Results

4.2.1 FewRel Benchmark

The accuracy of AGCKD for each task on Fewrel's 10-way 5-shot, 2-shot and 10-shot is described in Table 1 and Fig. 3. From these results, we can observe that:

(1) By comparing the mean in Table 1, we can find that AGCKD is significantly higher than the traditional method at each time step. The performance in T^1 indicates that AGCKD can effectively adopt the general knowledge learned by BERT. Meanwhile, AGCKD also achieves state-of-the-art performance in T^8, reflecting that the interference between different task parameters in AGCKD is the least, which mainly benefits from adaptive correction of the gradient and efficient decomposition of knowledge during model training. As a traditional method of CRE, CEAR requires many training data to learn the memory and embedding of a new task. When there is less training data, the features learned by the model will become unstable, which makes it collapse with the performance. ConPL and ERDA adjust the consistency of embedding indirectly only from the perspective of the loss function and data augmentation, which has limited ability to improve following degree. We further discuss the forgetting metrics of ConPL, ERDA and AGCKD in Appendix C.

(2) The standard deviation of AGCKD on the final task T^8 remains at a low level compared with the other methods, which indicates that it has a small fluctuation range when faced with different task sequences. It further reflects the robustness of AGCKD under different task sequences, thanks to the gradient correction matrix constraining the off-

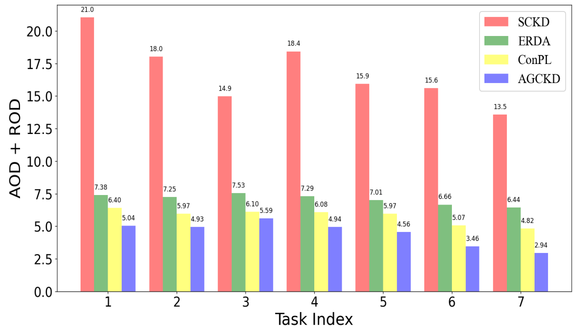

Figure 4: The sum of AOD and ROD of baselines on FewRel's 10-way 5-shot. The abscissa represents all 7 task indexes of models.

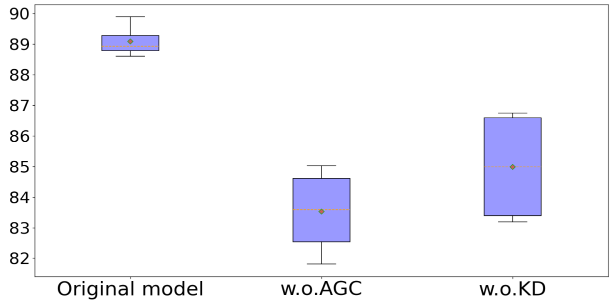

Figure 5: Box line diagram of ablation study on FewRel's 10-way-5-shot. The vertical axis represents accuracy of the model on the final task T^8.

set of the previous samples' embedding during parameter update. To prove this, we calculate the sum of AOD and ROD for each task in the model and show them in Fig. 4. It can be seen that AGCKD has low-level AOD and ROD on all tasks. In combination with Table 1, we observe that the smaller the sum of AOD and ROD of the model, the higher the accuracy. This is also consistent with our previous theoretical analysis of the following degree and offset distance.

(3) As is shown in Fig. 3, the performance of AGCKD is close to Joint Training, which indicates that AGCKD is less affected by catastrophic forgetting during training. In this case, we consider that more information introduced by new tasks also unexpectedly leads to the decline of model performance. Because the training times of samples at different time steps may be uneven during joint training, AGCKD even exceeds Joint Training on some tasks.

4.2.2 TACRED Benchmark

The performance of AGCKD for each task on TACRED's 5-way 5-shot and 10-shot is shown in Table 1 and Fig. 3. It can be observed from these results that AGCKD still has advantages over state-of-the-art methods. AGCKD has a strong generalization ability, which depends on our approach without any dataset-specific components.

4.3 Ablation Study

To verify the effectiveness of each part in AGCKD, we performed ablation experiments. Specifically, we separately remove adaptive gradient correction (w.o.AGC) and knowledge decomposition (w.o.KD). Average ablation results for the final tasks are presented in Fig. 5.

Through the box line diagram, the adaptive gradient correction and knowledge deconstruction module have a great impact on the average accuracy and performance stability of the model. The separate use of these two modules leads to confusion about the corresponding part of knowledge, which greatly damages the performance of the model. According to the definition, while reducing AOD, it also indirectly reduces ROD. Thus, the adaptive gradient correction has a greater impact on the final result. This phenomenon again demonstrates the effectiveness of the adaptive gradient correction algorithm and the importance of reducing the impact of subsequent tasks on previous tasks in the embedding space.

5 Conclusion

In this paper, a method of direct decoupling parameters and modifying gradient is proposed to improve the following degree of samples, which can eventually reduce the catastrophic forgetting for CFRE. Specifically, we first propose the concept of the following degree and analyze it from the perspectives of AOD and ROD. The parameters of the model are decomposed into general and task-related knowledge based on metric learning. For general knowledge, an adaptive gradient correction algorithm is proposed to reduce the impact of gradient updates on previous knowledge, which can reduce the AOD of samples. For task-related knowledge, we update the parameters discretely and continuously in three different training stages to optimize ROD between prototypes and samples. The theoretical derivation and experimental results on two standard benchmarks verify the superiority of AGCKD.

Limitations

In practical calculations, a small ROD is a sufficient and unnecessary condition for better model performance. Specifically, if ROD is large and the distance between prototypes is also relatively large, the model performance will not also be poor. In future work, we hope to consider the distance between classes in ROD and obtain a necessary and sufficient condition for model performance.

Currently, in the field of NLP, since the transformer-based model is the most widely used language model, we have only explored gradient correction algorithms for the relevant structures in the transformer. The performance of AGCKD in CFRE based on a transformer shows us the potential for designing gradient correction algorithms in other model architectures. And we will further explore the algorithm for gradient correction of parameters in other network structures (such as CNN, RNN, etc.).

References

Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. 2021. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 9516–9525.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2018. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*.

Xiudi Chen, Hui Wu, and Xiaodong Shi. 2023. Consistent prototype learning for few-shot continual relation extraction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7409–7422.

Zhiyuan Chen and Bing Liu. 2018. Lifelong machine learning . morgan & claypool publishers.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *ECAI 2020*, pages 2006–2013. IOS Press.

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

Yiduo Guo, Wenpeng Hu, Dongyan Zhao, and Bing Liu. 2022. Adaptive orthogonal projection for batch and online continual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6783–6791.

Xu He and Herbert Jaeger. 2018. Overcoming catastrophic interference using conceptor-aided backpropagation. In *International Conference on Learning Representations*.

Mahmut Kaya and Hasan Şakir Bilge. 2019. Deep metric learning: A survey. *Symmetry*, 11(9):1066.

Xianming Li, Xiaotian Luo, Chenghao Dong, Daichuan Yang, Beidi Luan, and Zhen He. 2021. Tdeer: An efficient translating decoding schema for joint extraction of entities and relations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8055–8064.

Yunyang Li, Zhinong Zhong, and Ning Jing. 2018. Multi-path convolutional neural network for distant supervised relation extraction. In *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*, pages 1–7.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Shuliang Liu, Xuming Hu, Chenwei Zhang, Lijie Wen, S Yu Philip, et al. 2022a. Hiure: Hierarchical exemplar contrastive learning for unsupervised relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5970–5980.

Yang Liu, Jinpeng Hu, Xiang Wan, and Tsung-Hui Chang. 2022b. A simple yet effective relation information guided approach for few-shot relation extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 757–763.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Zara Nasar, Syed Waqar Jaffry, and Muhammad Kamran Malik. 2021. Named entity recognition and relation extraction: State-of-the-art. *ACM Computing Surveys (CSUR)*, 54(1):1–39.

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.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. 2018. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pages 4528–4537. PMLR.

Yu-Ming Shang, Heyan Huang, and Xianling Mao. 2022. Onerel: Joint entity and relation extraction with one module in one step. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11285–11293.

Yisheng Song, Ting Wang, Puyu Cai, Subrota K Mondal, and Jyoti Prakash Sahoo. 2023. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Computing Surveys*.

Liyuan Wang, Xingxing Zhang, Qian Li, Jun Zhu, and Yi Zhong. 2022a. Coscl: Cooperation of small continual learners is stronger than a big one. In *European Conference on Computer Vision*, pages 254–271. Springer.

Xinyi Wang, Zitao Wang, and Wei Hu. 2023. Serial contrastive knowledge distillation for continual few-shot relation extraction. *arXiv preprint arXiv:2305.06616*.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022b. Promda: Prompt-based data augmentation for low-resource nlu tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4255.

Zhen Wang, Liu Liu, Yiqun Duan, and Dacheng Tao. 2022c. Continual learning through retrieval and imagination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8594–8602.

Heming Xia, Peiyi Wang, Tianyu Liu, Binghuai Lin, Yunbo Cao, and Zhifang Sui. 2023. Enhancing continual relation extraction via classifier decomposition. *arXiv preprint arXiv:2305.04636*.

Shipeng Yan, Lanqing Hong, Hang Xu, Jianhua Han, Tinne Tuytelaars, Zhenguo Li, and Xuming He. 2022. Generative negative text replay for continual vision-language pretraining. In *European Conference on Computer Vision*, pages 22–38. Springer.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. 2019. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372.

Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. 2019. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2759–2768.

Baoquan Zhang, Xutao Li, Shanshan Feng, Yunming Ye, and Rui Ye. 2022. Metanode: Prototype optimization as a neural ode for few-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9014–9021.

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

Wenzheng Zhao, Yuanning Cui, and Wei Hu. 2023. Improving continual relation extraction by distinguishing analogous semantics. *arXiv preprint arXiv:2305.06620*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033.

Wenxuan Zhou, Sheng Zhang, Tristan Naumann, Muhao Chen, and Hoifung Poon. 2022. Continual contrastive finetuning improves low-resource relation extraction. *arXiv preprint arXiv:2212.10823*.

A Mathematical Proof of Adaptive Gradient Correction Algorithm

To prove that the algorithm can adaptively correct the gradient so that the model updates have less impact on the previous task samples, we record $H_i^l \in R^{d\times 1}$ as the layer l embedding corresponding to the memory sample of category i and $E_i^l \in R^{d\times 1}$ as that corresponding to any training sample of category i. Since the memory samples are calculated according to the maximum cosine similarity of the average class sample, the vector angle of H_i^l and E_i^l is much smaller than that of $H_{j(j\neq i)}^l$ and E_i^l (also thanks to the optimization of metric learning). C_i^l is essentially a linear transformation of any vector into the space which is orthogonal to H_i^l. Accordingly, when the angle between H_i^l and E_i^l is small, E_i^l is approximately equal to zero vector after linear transformation through C_i^l. From a single vector to the entire C^l matrix, we can get:$\Delta W^l E^l = \frac{\partial L_{gen}}{\partial W^l}(C^l E^l) \approx 0$.

For the FFN layer in BERT, it avoids the interference of the gradient update of subsequent tasks on the embeddings of previous tasks:

$$\begin{aligned} Out_k^{FFN} &= X + W^{FFN}X \\ &= X + (W^{FFN} + \gamma \frac{\partial L_{gen}}{\partial W^{FFN}} C^{FFN})X \\ &= Out_{k+1}^{FFN} \end{aligned} \tag{11}$$

where Out_k^{FFN} is the output of FFN at time step k, $X \in R^{d\times 1}$ is $[MASK]$ token embedding of any previous sample, and C^{FFN} is calculated by memory $\hat{M}^k$.

For the self-attention module in BERT, we freeze the key and value matrices, while the gradient correction and parameter update are only performed on the query matrix. The common self-attention formula is as follows:

$$\begin{aligned} Q &= X^T W^q \\ K &= X^T W^k \\ V &= X^T W^v \end{aligned}$$
$$SelfAttention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d}})V \tag{12}$$

where $X = [x_1, \cdots, x_{[mask]}, \cdots, x_n] \in R^{d\times n}$ is the token embedding sequence of any previous sample. We set the updated Q as Q':

$$\begin{aligned} Q' =& X^T(W^q + \gamma (C^q)^T \frac{\partial L_{gen}}{\partial W^q}) \\ =& [q_1', \cdots, q_{[mask]}, \cdots, q_n']^T \end{aligned} \tag{13}$$

where vector $q_n' \in R^{1\times d}$ is the nth row of matrix Q'.

$$Q'K^T = \begin{bmatrix} q_1'k_1^T & q_1'k_2^T & \cdots & q_1'k_n^T \\ \vdots & \vdots & \ddots & \vdots \\ q_{[mask]}k_1^T & q_{[mask]}k_2^T & \cdots & q_{[mask]}k_n^T \\ \vdots & \vdots & \ddots & \vdots \\ q_n'k_1^T & q_n'k_2^T & \cdots & q_n'k_n^T \end{bmatrix} \tag{14}$$

Let $A' = softmax(\frac{Q'K^T}{\sqrt{d}})$, then the row vector corresponding to $[MASK]$ token in matrix A' is consistent with that in A, that is, $A' = [a_1', \cdots, a_{mask}, \cdots, a_n']^T$. Thus, the embedding of $[MASK]$ token in $A'V$ is consistent with that before the gradient update, i.e. the output of self-attention related to previous tasks is not affected by the updated gradient of subsequent tasks. In other words, our gradient correction method can also avoid the knowledge coverage caused by gradient updates in the self-attention layer.

B Training Details

The BERT-base (Devlin et al., 2018) is used as the encoder and is trained using AdamW (Loshchilov and Hutter, 2018) optimizer at a learning rate $\gamma = 2e-5$. The head of the attention mechanism is set to 8 and the embedding size of BERT is 768, i.e. $d = 768$. The batch size is set to 5. We train the model one time at the first training step ($epoch_1 = 1$), and three times at the second step and third step ($epoch_2 = epoch_3 = 1$). We set $\alpha = 0.2$ in P^{sim}. The memory is size set to 1 for each class. AGCKD can complete all tasks in about 20 minutes using one NVIDIA 4060Ti GPU. We run the model 6 times with different task sequences and report the mean result.

C Forgetting Metrics

The degree of catastrophic forgetting of AGCKD is measured by the forgetting metrics proposed by Chaudhry et al. (2018). After learning all tasks, the

Method	Task index							
	1	2	3	4	5	6	7	Mean
Joint Train	7.07	4.28	3.02	2.00	1.43	3.99	0.50	3.18
SeqRun	49.28	49.08	44.18	39.69	28.93	11.14	7.29	32.80
ERDA	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
AGCKD (Ours)	**7.87**	**5.34**	5.39	**2.40**	2.46	**−1.05**	0.30	**3.24**

Table 2: Forgetting (%) of various methods after training for each task on FewRel's 10-way-5-shot.

Method	Task index							
	1	2	3	4	5	6	7	8
Related to T^1	2.50	2.40	2.40	1.90	2.10	1.90	2.00	1.60
Related to $\hat{T}^k$	2.50	4.50	6.40	9.40	11.60	12.20	12.60	16.60

Table 3: The error rate (%) of AGCKD for each task on D^1_{test}. After the learning of T^k, the first line represents the error rate related to T^1; the second line represents that related to all tasks visible to the model.

following formula is calculated on all test datasets:

$$F_k = \frac{1}{n-k} \sum_{j=k+1}^{n} \max_{l \in \{k,\dots,j-1\}} (a_{l,k} - a_{j,k}) \quad (15)$$

where $a_{l,k}$ is the accuracy of the model on task k after the training step l and n is the total number of tasks. $F_k(k \in [1, \cdots, n-1])$ is the forgetting metrics of task k after all training steps. Obviously, the smaller F_k, the less knowledge the model forgets.

Table 2 shows the forgetting metrics after training for each task on FewRel's 10-way-5-shot. The average forgetting metrics of AGCKD are lower than that of other methods and are closest to Joint Training, which indicates that AGCKD can reduce catastrophic forgetting to a certain extent. At the same time, we find that even if all the training data are available, forgetting still exists. This may be because this index does not fully consider the information increment of the newly introduced task.

To observe the forgetting metrics more intuitively, Table 3 directly shows the total error rate and that related to T^1 at each time step k on D^1_{test}. The errors related to the first learning task of the model almost do not continue to increase, which intuitively reflects that AGCKD can effectively avoid the catastrophic forgetting of previous tasks.