Online Noisy Continual Relation Learning

Guozheng Li[1], **Peng Wang**[1*], **Qiqing Luo**[1], **Yanhe Liu**[1], **Wenjun Ke**[1,2]
[1]School of Computer Science and Engineering, Southeast University
[2]Beijing Institute of Computer Technology and Application
{gzli, pwang, qqluo, liuyanhe}@seu.edu.cn, kewenjun2191@163.com

Abstract

Recent work for continual relation learning has achieved remarkable progress. However, most existing methods only focus on tackling catastrophic forgetting to improve performance in the existing setup, while continually learning relations in the real-world must overcome many other challenges. One is that the data possibly comes in an online streaming fashion with data distributions gradually changing and without distinct task boundaries. Another is that noisy labels are inevitable in real-world, as relation samples may be contaminated by label inconsistencies or labeled with distant supervision. In this work, therefore, we propose a novel continual relation learning framework that simultaneously addresses both online and noisy relation learning challenges. Our framework contains three key modules: (i) a sample separated online purifying module that divides the online data stream into clean and noisy samples, (ii) a self-supervised online learning module that circumvents inferior training signals caused by noisy data, and (iii) a semi-supervised offline finetuning module that ensures the participation of both clean and noisy samples. Experimental results on FewRel, TACRED and NYT-H with real-world noise demonstrate that our framework greatly outperforms the combinations of the state-of-the-art online continual learning and noisy label learning methods.

Introduction

Relation extraction (RE), aiming to identify the relationships between entities in texts, is an essential task in information extraction (IE), which can apply to many downstream natural language processing (NLP) tasks, such as information retrieval and question answering. However, traditional RE methods (Zeng et al. 2014; Zhang et al. 2017; Baldini Soares et al. 2019; Zhu et al. 2020; Li et al. 2022) always assume the relations are predefined and fixed, which cannot handle the new relations emerging in the real-world scenarios. Hence, the ability to continually learn relations from sequential data has gained much attention in recent RE research (Wang et al. 2019a; Han et al. 2020; Cui et al. 2021; Zhao et al. 2022), which is often coined as continual relation extraction (CRE). Despite rapid advances in CRE, most of recent researches are devoted to improving performance in the existing CRE setup. However, the existing setup still lacks practicality in certain aspects.

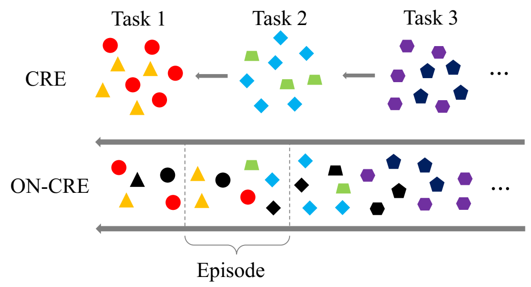

Figure 1: Differences between CRE and ON-CRE. CRE is offline and task-based with whole clean data, whereas ON-CRE is online and task-free with falsely labeled data (black samples). An episode holds the samples only allowed to use once for parameter update during a specific time period.

On the one hand, it is an offline task-based sequential learning setup, which means that a sequence of tasks are learned, one at a time, with all data of current task available but not of previous or future tasks. In other words, the model knows the task boundaries at the training time and the CRE process is divided perfectly with separate tasks. This setup, however, is rarely encountered in practical applications. Therefore, we propose an online task-free CRE setup that keeps on learning relations over time in a streaming fashion, using each streamed sample only once, with data distributions gradually changing and without the notion of separate tasks. Figure 1 shows the differences between CRE and our proposed setup. The samples in an episode could come from different tasks. Unfortunately, existing methods (Wang et al. 2019a; Han et al. 2020; Cui et al. 2021; Zhao et al. 2022) cannot be applied in online task-free CRE scenarios as they are all trained in the offline paradigm.

On the other hand, previous works assume that all the samples are labeled correctly. However, learning from data riddled with noisy labels is an inevitable scenario in real-world especially in the context of RE. Supervised RE models suffer from the lack of large-scale high-quality training data, since manually labeling data is time-consuming and human-intensive. To alleviate this issue, distant supervision (Mintz et al. 2009) has been used to automatically label data by

[*]Corresponding author.

aligning existing knowledge bases (KBs) with plain texts. Obviously, such automatic labeling mechanism inevitably brings noisy labels, because an entity pair may not express the relation from KBs in a specific sentence. Therefore, considering the noisy relations in CRE is quite necessary, which is beneficial for more realistic CRE in real-world.

To this end, we aim to jointly tackle the problems of online and noisy relation learning in CRE, which to the best of our knowledge have not been studied in prior work. We leverage the replay-based continual learning approach to handle this new scenario, while previous works have proved that replay-based methods are the most effective in NLP applications (de Masson d'Autume et al. 2019) including RE (Wang et al. 2019a). Nonetheless, replaying a noisy memory will aggravate the forgetting process due to the unreliable mapping of previously acquired knowledge (Kim et al. 2021), which indicates that maintaining a clean replay buffer is extremely important. Moreover, existing noisy label learning methods suffer from handling the online task-free setting (Rolnick et al. 2019; Aljundi et al. 2019; Prabhu, Torr, and Dokania 2020) because they assume that the whole dataset is given to purify the noise. Therefore, we propose a purification technique that could divide the online task-free data stream into clean and noisy samples. Empirical results by Arpit et al. (2017) show that deep neural networks learn the simple pattern first before memorizing the noisy labels through long-term training, which is consistent with the requirements of online learning as each streamed sample is used only once. Specifically, we train a purifier in online manner and separate samples based on the mean square error (MSE) loss (Karim et al. 2022) because clean samples tend to have lower loss compared to noisy ones.

Although the purifier can separate the samples, directly dropping the noisy samples filtered by purifier is infeasible because the noise rate may be high which signifies plenty of data is wasted. However, learning with noisy labeled data could lead to incorrect back propagation signals. Therefore, we utilize all the streaming data at online learning stage to alleviate noisy errors using self-supervised contrastive learning (Chen et al. 2020), which treats all the data as unlabeled data and learns general relation representations. At inference, we perform finetuning in a semi-supervised fashion via conditional entropy minimization (Grandvalet and Bengio 2004) using all the samples stored in the memory including clean and noisy samples. Taking noisy data as unlabeled data and incorporating its unsupervised features improve the overall performance significantly.

In this paper, we propose a simple continual relation learning framework to simultaneously address both online and noisy relation learning challenges, which contains **S**ample **S**eparated online purifying, **S**elf-**S**upervised online learning and **S**emi-**S**upervised offline finetuning (**S6**). Specifically, we first separate the delay buffer storing the incoming data stream into a clean delay buffer and a noisy delay buffer. If the clean or noisy delay buffer is full, specific samples are stored in corresponding replay buffer. Simultaneously, an online learner is trained in a self-supervised fashion with data in replay and delay buffer. At test time, we copy the parameters of the learner and finetune on clean and noisy replay buffers in a semi-supervised fashion. In summary, the main contributions of this paper are three-fold:

- To the best of our knowledge, we are among the first to consider **O**nline and **N**oisy learning scenarios in the realm of CRE, which is a more realistic setup called **ON**-CRE. We define the problem and construct three benchmarks for this problem.
- We propose S6, a novel continual relation learning framework for ON-CRE based on sample separated online purifying, self-supervised online learning and semi-supervised offline finetuning.
- Experimental results demonstrate that S6 outperforms current state-of-the-art on two synthetic noise benchmarks of FewRel and TACRED, and one real-world noise benchmark of NYT-H.

Related Work

Continual Learning Existing continual learning mainly focuses on three aspects: (1) Regularization-based methods (Zenke, Poole, and Ganguli 2017) update important parameters with constraints to relieve catastrophic forgetting. (2) Dynamic architecture methods (Chen, Goodfellow, and Shlens 2016) extend the model architecture to learn new tasks, so the model size increases continually. (3) Memory-based methods (de Masson d'Autume et al. 2019) save some samples from old tasks and replay them back in new tasks, which have been proved to be the most effective method in NLP. In CRE, Wang et al. (2019a) utilizes an explicit alignment model to mitigate the sentence embedding distortion when encountering new relations. Han et al. (2020) introduces a joint training process for memory activation and consolidation. Cui et al. (2021) tries to refine the relation prototypes to better recover the interruption of the embedded space. And Zhao et al. (2022) incorporates both contrastive learning and knowledge distillation to maintain the stability of the relation embedding. But these methods cannot be applied to ON-CRE as they are all trained in offline scenarios.

Online Learning In the online continual learning scenario, even though the learner is allowed to store samples as they come, it can only use a training sample (unless it is in the memory) once to update its parameters (Aljundi et al. 2019). On the contrary, the offline setting allows unrestricted access to the entire dataset corresponding to a specific task, and one can revisit the samples multiple times during the learning process (Rebuffi et al. 2017). There are few studies in online RE. Peng et al. (2019) adopts an online clustering algorithm to extract potential semantic centers of each relation. Wang et al. (2022) deploys a caching updater to refresh the relation representations at each training step in an online manner. However, online CRE is not studied in prior work.

Noisy Label Learning Learning with noisy labeled data has been a long-studied problem. Recent work (Wang et al. 2019b; Kun and Jianxin 2019; Wei et al. 2020) tries to overcome this issue by regularizing, repairing, filtering, etc. However, these methods are unsuitable in online continual learning setting because they require plenty of samples to work. Distantly supervised RE (Mintz et al. 2009) needs to consider the negative impact of noisy relations, but they are different from the sentence-level RE because they assume several samples with same relation stored in a bag which is also coined as bag-level RE. Neither noisy label learning nor distantly supervised RE methods are suitable for ON-CRE.

Methodology

Problem Statements

We consider the scenario of online noisy continual relation learning. A sample (x_t, y_t), where x_i is the input data including the natural language text and entity pair, and $y_t \in \mathcal{R}$ is the relation label, arrives at each time step t and the data distribution is gradually changing without task identifiers. Moreover, we assume that some of the data is incorrectly labeled. The goal of ON-CRE is to keep learning new tasks while avoiding catastrophic forgetting of previous tasks in both online and noisy learning settings.

Model Overview

The overview of S6 is illustrated in Figure 2, which contains three key components: sample separated online purifying, self-supervised online learning and semi-supervised offline finetuning. In ON-CRE, a delay buffer $\mathcal{D}$ of limited size is used to store the incoming data and our first objective is to separate this buffer into a clean delay buffer $\mathcal{C}$ and a noisy delay buffer $\mathcal{N}$. Whenever $\mathcal{C}$ or $\mathcal{N}$ is full, we put several samples into clean replay buffer $\mathcal{M}_\mathcal{C}$ or noisy replay buffer $\mathcal{M}_\mathcal{N}$. First, we train a purifier consisting of an encoder $f_\phi(\cdot)$ and an online classifier on $\mathcal{D}$. Then we adopt MSE loss to determine a threshold and separate $\mathcal{D}$ into $\mathcal{C}$ and $\mathcal{N}$. Simultaneously, we also define a learner $f_\theta(\cdot)$ and train it on $\mathcal{D}$ via self-supervised contrastive loss. Moreover, we perform a replay on $\mathcal{M}_\mathcal{C}$ and $\mathcal{M}_\mathcal{N}$ to relieve catastrophic forgetting in $f_\theta(\cdot)$. At inference time, we copy the parameters of learner $f_\theta(\cdot)$ and add a relation classifier to finetune on $\mathcal{M}_\mathcal{C}$ and $\mathcal{M}_\mathcal{N}$ in a semi-supervised fashion.

Encoder

For the input relation sample (x, y), we utilize BERT (Devlin et al. 2019) as the backbone of $f_\phi(\cdot)$ and $f_\theta(\cdot)$ to encode entity pairs and context information to get the relation representation. Following (Baldini Soares et al. 2019), given a sample x including a sentence and its entity pair (E1, E2), we augment x with four special tokens to indicate the begin and end of each entity mentioned in the sentence. Then the augmented sentence $\tilde{x}$ is fed to BERT and the output corresponding to the positions of E1 and E2 are concatenated to map to a hidden representation $\mathbf{h} \in \mathbb{R}^d$ as follows:

$$\mathbf{h} = \mathbf{W}[\mathbf{h}_{[\text{E1}]}; \mathbf{h}_{[\text{E2}]}] + \mathbf{b} = \mathbf{W}[\text{BERT}(\tilde{x})] + \mathbf{b} \quad (1)$$

where $\mathbf{W} \in \mathbb{R}^{d\times 2d}$ and $\mathbf{b} \in \mathbb{R}^d$ are trainable parameters. And $\mathbf{h}$ serves as the input feature for online classifier, projection head and relation classifier.

Sample Separated Online Purifying

Deep neural networks learn the simple pattern first before memorizing the noisy labels through long-term training, which means that clean samples tend to have lower loss compared to noisy samples in brief period of online training. Therefore, we train a temporary purifier (denoted as $f_\phi(\cdot)$ below for symbol convenience) and set an appropriate separation threshold during each online learning episode. Specifically, whenever the delay buffer $\mathcal{D}$ is full, we perform supervised learning on $\mathcal{D}$ to obtain a purifier $f_\phi(\cdot)$:

$$\mathcal{L}_p = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} -y \log f_\phi(y|x) \quad (2)$$

For every x_i, its prediction probabilities obtained by purifier $f_\phi(\cdot)$ is denoted as $\mathbf{p}_i = [p_i^1, p_i^2, ..., p_i^n]$, where n is the size of relation set $\mathcal{R}_\mathcal{D}$ in $\mathcal{D}$. Similarly, the ground-truth label distribution for x_i is $\mathbf{y}_i = [y_i^1, y_i^2, ..., y_i^n]$, where $y_i^j \in \{0, 1\}$. We adopt mean square error (MSE) loss (Karim et al. 2022) to measure the differences l_i between $\mathbf{p}_i$ and $\mathbf{y}_i$. Then we estimate the separation threshold l_t for clean and noisy samples as the mean of loss distribution $\mathbf{l}_\mathcal{D} = [l_1, l_2, ..., l_{|\mathcal{D}|}]$:

$$l_t = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \text{MSE}(\mathbf{p}_i, \mathbf{y}_i) = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{n}(p_i^j - y_i^j)^2 \quad (3)$$

Based on the separation threshold, the samples with loss lower than l_t are sent to clean delay buffer $\mathcal{C}$ and the rest ones are sent to noisy delay buffer $\mathcal{N}$. This sample separation process corresponds to the line 3 $\sim$ 19 in Algorithm 1.

Whenever $\mathcal{C}$ or $\mathcal{N}$ is full, we perform sample selection to store several samples in replay buffer $\mathcal{M}_\mathcal{C}$ or $\mathcal{M}_\mathcal{N}$. We stipulate that $\mathbf{l}_\mathcal{C}$ and $\mathbf{l}_\mathcal{N}$ are loss distribution containing loss values for samples in $\mathcal{C}$ and $\mathcal{N}$, respectively. Then we let $\mathbf{l}_\mathcal{C}^{r_j}[K_1]$ indicate that it only contains the lowest K_1 loss values for samples with relation $r_j \in \mathcal{R}_\mathcal{D}$ in $\mathbf{l}_\mathcal{C}$, while $\mathbf{l}_\mathcal{N}^{r_j}[K_2]$ only contains the highest K_2 loss values for samples with relation $r_j \in \mathcal{R}_\mathcal{D}$ in $\mathbf{l}_\mathcal{N}$. The select rules for the replay buffers are expressed as:

$$\mathcal{M}_\mathcal{C} \leftarrow \mathcal{M}_\mathcal{C} \cup \{(x_i, y_i) : \forall l_i \in \mathbf{l}_\mathcal{C}^{r_j}[K_1], r_j \in \mathcal{R}_\mathcal{D}\} \quad (4)$$

$$\mathcal{M}_\mathcal{N} \leftarrow \mathcal{M}_\mathcal{N} \cup \{(x_i, y_i) : \forall l_i \in \mathbf{l}_\mathcal{N}^{r_j}[K_2], r_j \in \mathcal{R}_\mathcal{D}\} \quad (5)$$

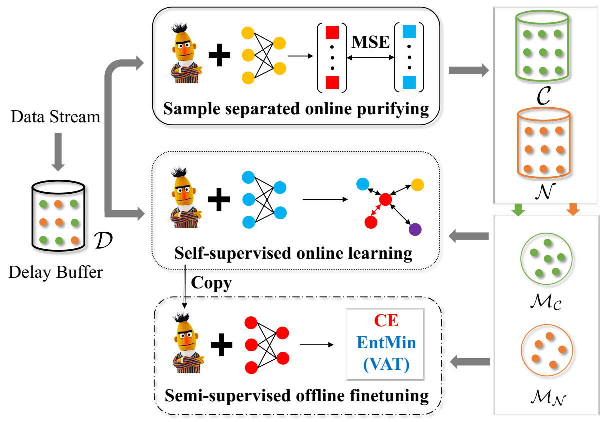
MSE
Sample separated online purifying
Data Stream
$\mathcal{C}$
$\mathcal{N}$
$\mathcal{D}$
Delay Buffer
Self-supervised online learning
Copy
$\mathcal{M}_\mathcal{C}$
CE
EntMin
(VAT)
Semi-supervised offline finetuning
$\mathcal{M}_\mathcal{N}$

Figure 2: Illustration of the S6 framework.

Algorithm 1: Online Purifying and Learning for S6

```
Input: Online streaming data $(x_1, y_1), ..., (x_n, y_n)$, and the
    learner $f_\theta(\cdot)$.
Output: The learner $f_\theta(\cdot)$, the clean replay buffer $\mathcal{M}_\mathcal{C}$ and the
    noisy replay buffer $\mathcal{M}_\mathcal{N}$.
 1: $\mathcal{D} = \mathcal{C} = \mathcal{N} = \mathcal{M}_\mathcal{C} = \mathcal{M}_\mathcal{N} = \varnothing$
 2: for $t \leftarrow 1$ to $n$ do
 3:   if $\mathcal{D}$ is full then
 4:     Initialize a purifier $f_\phi(\cdot)$ for current $\mathcal{D}$
 5:     for minibatch $\mathcal{B} \in \mathcal{D} \cup \mathcal{M}_\mathcal{C} \cup \mathcal{M}_\mathcal{N}$ do
 6:       $\mathcal{B}_\phi = \{(x, y) : (x, y) \in \mathcal{B} \cap \mathcal{D}\}$
 7:       Update $f_\phi(\cdot)$ on $\mathcal{B}_\phi$ based on Equation 2
 8:       Update $f_\theta(\cdot)$ on $\mathcal{B}$ based on Equation 6
 9:     end for
10:     Calculate the threshold $l_t$ based on Equation 3
11:     for $i \leftarrow 1$ to $|\mathcal{D}|$ do
12:       $l_i = \mathrm{MSE}(f_\phi(x_i), \mathrm{OneHot}(y_i))$
13:     end for
14:     $\mathcal{C} \leftarrow \mathcal{C} \cup \{(x_i, y_i) : \forall\, l_i < l_t\}$
15:     $\mathcal{N} \leftarrow \mathcal{N} \cup \{(x_i, y_i) : \forall\, l_t \leq l_i\}$
16:     Reset $\mathcal{D}$
17:   else
18:     $\mathcal{D} \leftarrow \mathcal{D} \cup (x_t, y_t)$
19:   end if
20:   if $\mathcal{C}$ is full then
21:     Update $\mathcal{M}_\mathcal{C}$ based on Equation 4
22:     Reset $\mathcal{C}$
23:   end if
24:   if $\mathcal{N}$ is full then
25:     Update $\mathcal{M}_\mathcal{N}$ based on Equation 5
26:     Reset $\mathcal{N}$
27:   end if
28: end for
29: return $f_\theta(\cdot), \mathcal{M}_\mathcal{C}, \mathcal{M}_\mathcal{N}$
```

In other words, we hold K_1 number of highly confident (with low loss values) clean samples with every relation from $\mathcal{C}$ in the clean replay buffer $\mathcal{M}_\mathcal{C}$. In contrast, K_2 number of noisy samples with high loss values are stored in the noisy replay buffer $\mathcal{M}_\mathcal{N}$. The sample selected process is described in line 20 $\sim$ 27 in Algorithm 1.

Self-supervised Online Learning

Through purification, we actually have separated clean samples and noisy samples. However, directly dropping the noisy samples is such a waste which may result in few clean samples for fully supervised learning. Hence, we alleviate this issue via learning only from x_t using self-supervised contrastive learning. To mitigate catastrophic forgetting, we treat all the data in current delay buffer and replay buffers as unlabeled data to learn general relation representations, and finetune the representation using only the samples in the replay buffers at inference time, which is similar with GDumb (Prabhu, Torr, and Dokania 2020).

Specifically, we add a projection head $g(\cdot)$ on top of the learner $f_\theta(\cdot)$, and train it using self-supervised contrastive loss. For a minibatch $\mathcal{B}$ from $\mathcal{D} \cup \mathcal{M}_\mathcal{C} \cup \mathcal{M}_\mathcal{N}$, we apply easy data augmentation (Wei and Zou 2019) (e.g., random insertion, random swap, random deletion) to create positive samples. Then we train it using the normalized temperature-scaled cross-entropy loss (Chen et al. 2020):

$$\mathcal{L}_s = -\sum_{i=1}^{2|\mathcal{B}|} \log \frac{e^{\mathbf{u}_i^\top \mathbf{u}_j / \tau}}{\sum_{k=1, k\neq i}^{2|\mathcal{B}|} e^{\mathbf{u}_i^\top \mathbf{u}_k / \tau}} \tag{6}$$

where $\mathbf{u}_z = \frac{g(f_\theta(x_z))}{||g(f_\theta(x_z))||_2}, z \in \{i, j, k\}$ denotes the L_2 normalized feature, and $\tau > 0$ is the temperature. The process of online learning is shown in line 5 $\sim$ 9 in Algorithm 1.

Semi-supervised Offline Finetuning

At any inference time, we finetune the online learner $f_\theta(\cdot)$ using only the samples in both clean and noisy replay buffers. Since the labels of the noisy replay buffer cannot be trusted, we consider them as unlabeled data. Therefore, we can define the labeled and unlabeled sets, $\mathcal{M}_\mathcal{C}$ and $\mathcal{M}_\mathcal{N}$, and finetune the learner $f_\theta(\cdot)$ in a semi-supervised fashion. The semi-supervised methods we consider in this paper have a learning objective of the following form:

$$\mathcal{L}_f = \min_\theta \mathcal{L}_l(\mathcal{M}_\mathcal{C}, \theta) + \omega \mathcal{L}_u(\mathcal{M}_\mathcal{N}, \theta) \tag{7}$$

where ω is a non-negative scalar weight and θ is the parameters for the learner $f_\theta(\cdot)$. $\mathcal{L}_l$ is a standard cross-entropy classification loss of all clean relation samples, and $\mathcal{L}_u$ is a loss defined on unsupervised relation samples. Here we adopt the conditional entropy minimization (EntMin) (Grandvalet and Bengio 2004) to encourage the model to make confident predictions on unlabeled data, because we assume that unlabeled data does have a class we are training on, even if it is unknown. Here we treat $f_\theta(\cdot)$ as a conditional distribution of relations over samples, and the loss is defined as:

$$\mathcal{L}_u = \frac{1}{|\mathcal{M}_\mathcal{N}|} \sum_{x \in \mathcal{M}_\mathcal{N}} \sum_{y \in \mathcal{R}_\mathcal{M}} -f_\theta(y|x) \log f_\theta(y|x) \tag{8}$$

where $\mathcal{R}_\mathcal{M}$ denotes the relation set in replay buffers. We should point out that many semi-supervised learning techniques could be applied to this fine-tuning stage, which will not affect our overall framework. For instance, the virtual adversarial training (VAT) (Miyato, Dai, and Goodfellow 2017) could also be used as:

$$\mathcal{L}_u = \frac{1}{|\mathcal{M}_\mathcal{N}|} \sum_{x \in \mathcal{M}_\mathcal{N}} \mathrm{KL}(f_\theta(x) \,||\, f_\theta(x + \Delta x)) \tag{9}$$

$$\Delta x = \arg\max_{|\delta|_2 = \epsilon} \mathrm{KL}(f_\theta(x) \,||\, f_\theta(x + \delta)) \tag{10}$$

where ϵ is a hyperparameter. The offline finetuning is described in Algorithm 2.

Algorithm 2: Offline Finetuing for S6

```
Input: Testing data $(x_1, y_1), ..., (x_n, y_n)$, the learner $f_\theta(\cdot)$, the
    clean and noisy replay buffers $\mathcal{M}_\mathcal{C}$ and $\mathcal{M}_\mathcal{N}$.
1: Copy $f_\theta(\cdot)$ for offline finetuing
2: Update $f_\theta(\cdot)$ on $\mathcal{M}_\mathcal{C} \cup \mathcal{M}_\mathcal{N}$ based on Equation 7
3: for $t \leftarrow 1$ to $n$ do
4:   Classify the relation of $x_t$ using $f_\theta(\cdot)$
5: end for
```

Experiments

In our evaluation, we compare S6 with other state-of-the-art models on the ON-CRE setup. We test on two datasets of FewRel (Han et al. 2018) and TACRED (Zhang et al. 2017) with synthetic noise. And we adopt the NYT-H (Zhu et al. 2020) dataset obtained by distant supervision (Mintz et al. 2009) to evaluate under the real-world noise.

Experimental Design

Setup We create a synthetic noisy labeled dataset from FewRel and TACRED. Namely, we assign $\{20\%, 40\%, 60\%, 80\%\}$ samples of the dataset to other relations within the dataset by a uniform probability. Following previous work (Han et al. 2020; Cui et al. 2021; Zhao et al. 2022), we create 10 tasks with 8 relations in each task by selecting random classes without replacement for FewRel, and 10 tasks with 4 relations in each task for TACRED. In NYT-H, we only use its test set (9,955 sentences in total) because it is manually labeled and we can know which sentences are labeled incorrectly via distant supervision. We randomly divide 7 tasks with 3 relations in each task. And for each relation, we use 50% of the clean samples as the test samples. The main differences between ON-CRE and CRE are: (i) the task boundaries are agnostic instead of available, (ii) the data of one episode instead of data of one task is accessed at a time, (iii) the data is allowed to use only once instead of multiple times for parameter update, and (iv) the data is partly noisy instead of completely pure. For metrics, we use the overall accuracy which calculates the accuracy on the test set of all tasks on FewRel and TACRED following (Han et al. 2020). And we adopt the macro-F1 on NYT-H because it is an extremely imbalanced dataset.

Baselines Since we aim to tackle the ON-CRE problem, we design the baselines combining existing state-of-the-art online task-free continual learning methods and noisy label learning methods. We do not compare with the existing CRE models (Wang et al. 2019a; Han et al. 2020; Cui et al. 2021; Zhao et al. 2022) because they are trained in offline scenarios and require the task boundaries. Therefore, we select three online continual learning methods (ER (Rolnick et al. 2019), MIR (Aljundi et al. 2019) and GDumb (Prabhu, Torr, and Dokania 2020)) and equip them with three noisy label learning methods (SL (Wang et al. 2019b), Pencil (Kun and Jianxin 2019) and JoCoR (Wei et al. 2020)) having robust loss, architecture or regularization. And we should point out that existing sample selection methods (Pleiss et al. 2020) are unable to deploy in online scenarios directly. Besides, Multitask stores all previous samples in memory, trains the model on all data for each new task and servers as an upper bound. Finetune serves as a lower bound since it performs online training on the new tasks without using any memory.

Implementation Details All continual learning methods use BERT (Devlin et al. 2019) as their backbones. And the size of memory for replaying is set to 1,600, 800, 400 for FewRel, TACRED and NYT-H, respectively. For noisy label learning methods, we use $\alpha = \beta = 1.0$ in SL, $\alpha = 0.1$, $\beta = 0.4$, $\lambda = 600$ in Pencil, and $\lambda = 0.1$ in JoCoR, respectively.

Datasets	**FewRel**				**TACRED**				**NYT-H**
Noise Rate (%)	20	40	60	80	20	40	60	80	64.63
Multitask	73.1	68.9	55.3	36.9	78.4	47.7	18.2	11.0	40.4
Finetune	14.9	14.0	12.7	7.6	15.5	12.9	8.5	3.7	8.7
ER	45.4	31.8	18.9	11.1	27.3	18.9	10.2	5.4	14.1
ER+SL	51.4	38.6	24.5	16.0	31.8	20.2	10.9	6.0	15.9
ER+Pencil	52.2	35.0	16.4	8.2	32.1	19.7	10.6	5.7	19.4
ER+JoCoR	45.6	32.0	20.9	13.5	27.0	19.0	12.8	8.6	14.2
MIR	44.8	34.7	24.1	16.2	27.3	19.0	10.0	5.3	12.5
MIR+SL	51.6	37.0	23.7	15.1	31.5	19.8	10.1	5.2	14.2
MIR+Pencil	54.6	26.1	13.8	7.4	32.0	20.8	10.8	5.6	16.2
MIR+JoCoR	46.9	34.2	22.7	15.1	27.3	18.8	10.3	5.6	13.8
GDumb	54.2	39.3	24.9	17.8	40.7	22.5	9.8	6.3	22.2
GDumb+SL	52.5	42.2	26.9	16.9	37.8	23.4	10.2	6.6	20.5
GDumb+Pencil	52.9	39.3	25.5	15.9	37.5	22.8	10.2	6.4	19.6
GDumb+JoCoR	54.2	43.3	25.9	15.2	36.7	21.4	9.1	5.8	22.4
S6 (Ours)	**66.1**	**66.0**	**58.8**	**50.1**	**61.2**	**43.9**	**24.3**	**12.2**	**29.1**

Table 1: Main results. Results of all the methods are the average of random five times.

For fair comparison, We need to ensure that S6 uses roughly the same size of replay buffer as others. For FewRel, we set the size of $\mathcal{D}, \mathcal{C}, \mathcal{N}$ as 1,000, 1,000 and 2,000. For TACRED and NYT-H, we set the size of $\mathcal{D}, \mathcal{C}, \mathcal{N}$ as 200, 200 and 400. The learning rates of $f_\phi(\cdot)$ and $f_\theta(\cdot)$ are 5e-6 and 5e-5. The update sizes K_1 and K_2 are both 5. The batch size is 16 for both online and offline training. And we train 60 epochs in the finetuning stage.

Main Results

According to the main results in Table 1, S6 performs the best in all synthetic noise types with different levels of 20%, 40%, 60% and 80% as well as real-world noise. Obviously, S6 significantly outperforms other baselines on FewRel, especially when the noise rate is high. With the increase of noise rate, the performances of other methods drop sharply, while S6 still maintains its performance. When the noise rate is 80%, S6 outperforms other baselines over 30%, which indicates the superiority of our method. In some cases especially the high noise scenario, the noise label learning methods are not very effective or even worse than vanilla baselines. The online learning holds an episode to store fewer samples but these methods require plenty of samples to estimate the optimizing dynamics and reduce the noise by regularizing or repairing. So our proposed purifying technique is more suitable in online scenarios.

However, S6 drops significantly like other baselines on TACRED as the noise increases. Because the samples in TACRED are imbalanced, purifying the data and training the learner are more challenging compared to balanced FewRel benchmark when the noise increasing. Nonetheless, our method still has obvious advantage over the existing methods on imbalanced benchmark. Note the performance gap between GDumb and S6 on NYT-H is not as big as the results on others. NYT-H is an extremely unbalanced dataset in which some relations have thousands of samples and others have only dozens or several samples. As will be shown

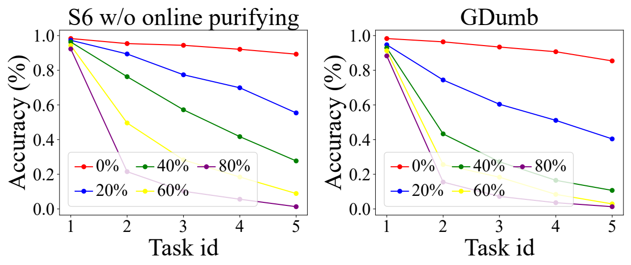

Figure 3: Impact of noisy data. We only replace the select rules illustrated in Equation 4 and 5 with random sampling on $\mathcal{D}$ keeping the same update size. In the first five tasks, the accuracy of the first task is plotted.

Datasets	**FewRel**				**TACRED**				**NYT-H**
Noise Rate (%)	20	40	60	80	20	40	60	80	64.63
Greedy Sampler	78.3	54.8	39.7	18.9	72.4	55.5	36.2	17.1	**36.3**
Sample Separator	**99.7**	**96.3**	**80.7**	**61.6**	**87.0**	**69.3**	**47.2**	**23.7**	34.5

Table 2: Purification of noisy data. Greedy sampler is proposed by GDumb and sample separator is our method.

later, the purifying technique is more suitable for the balanced scenario, and this is also the reason why S6 performs well on FewRel with high noise rate but not very well on TACRED. Actually, the performance gain on NYT-H is mainly the contribution of self-supervised online learning and semi-supervised offline finetuning. And we will demonstrate the effectiveness of proposed three key components later.

Analysis of Online Purifying

We conduct extensive experiments to verify the importance and effectiveness of online purifying. We notice that the quality of samples in clean replay buffer is essential for relieving catastrophic forgetting. We also highlight the superiority of proposed sample separator.

- **Impact of Noisy Data** To find the impact of noisy samples in the replay buffer, we compare the performance of S6 without online purifying module and GDumb on FewRel with different noise rates, and the results are illustrated in Figure 3. It shows that the noisy relations accelerate catastrophic forgetting. When encountering more noise, significantly hastened forgetting processes appear, while the forgetting curves of S6 and GDumb change very gently on the clean (0% noise) dataset. Therefore, we believe that an ideal replay buffer should maintain clean and shield the model from noise samples.
- **Purification of Noisy Data** We examine the proportion of clean samples in the clean replay buffer on three benchmarks, as shown in Tabel 2. The sample separator keeps the proportion of clean samples at a high level on FewRel with 20% and 40% noise rate. Even with 80% noise, it could still filter almost 40% of the noisy samples in the clean replay buffer, which shows the effectiveness of our purifying technique. However, it suffers grave declines encountering with imbalanced datasets like TACRED and NYT-H. If the imbalance is not serious and the noise rate is not high, our method still filters the noisy samples effectively. Hence, our method shows great power of purifying the balanced dataset but performs modestly effective on imbalanced ones. On NYT-H, it is even worse than greedy sampler as the number of samples is extremely imbalanced. In this way, all the samples corresponding to some rare relations are actually not divided into clean delay buffer because of the common relations with excessive samples.

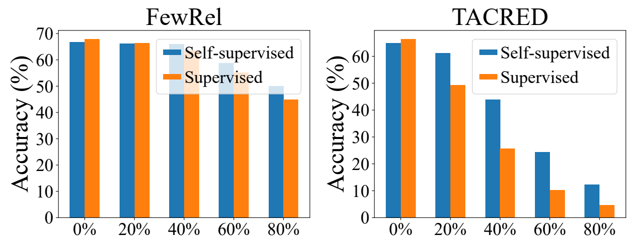

Figure 4: Self-supervised and supervised comparison. We replace the loss function illustrated in Equation 6 with supervised contrastive learning loss (Khosla et al. 2020).

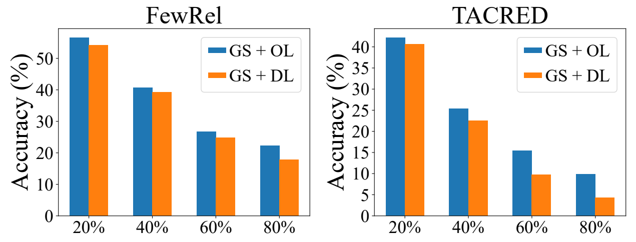

Figure 5: Online learner improves GDumb. GS + OL denotes the greedy sampler combining online learner, while GS + DL is the original GDumb.

Effectiveness of Online Learning

We conduct two experiments to demonstrate the advantages of self-supervised online learning. We first show the self-supervised learning circumvents inferior training signals caused by noisy signals compared to supervised learning. Then we verify the effectiveness of our online learner.

- **Self-supervised and supervised comparison** The results of self-supervised contrastive learning versus supervised contrastive learning are shown in Figure 4. Overall, our method outperforms the supervised contrastive learning paradigm. And the performance gap increases as the noise rate increases, which is consistent with intuition. When the training data is dominated by noisy samples, the supervised contrastive loss is misled by the wrong training signals, so as to degrade the overall performance significantly. Nonetheless, the performance of supervised fashion is close or even slightly better under the low noise rate or clean scenario, because the clean labels provide more accurate optimization directions.
- **Online learner improves GDumb** The learner obtained by self-supervised online learning can replace the dumb learner in GDumb, which results in better performance as

Replay Buffer	**FewRel**			**TACRED**			**NYT-H**		
	Num.	Acc.	Cov.	Num.	Acc.	Cov.	Num.	Acc.	Cov.
Only Clean	**1,171**	49.1	✓	**460**	10.5	✓	**168**	18.2	13 / 21
Twice Clean	2,330	49.6	✓	902	10.8	✓	312	20.4	13 / 21
Noisy & Clean	1,946	**50.1**	✓	795	**12.2**	✓	324	**29.1**	✓

Table 3: Offline finetuning results on FewRel, TACRED and NYT-H. Only Clean merely utilizes the clean replay buffer. Twice Clean doubles the capacity of the clean replay buffer, while Noisy & Clean utilizes both noisy and clean replay buffers. Num. denotes the number of samples in the replay buffers. Acc. denotes the overall accuracy (%) of different methods. And Cov. means the relations in the replay buffers whether cover the whole relation set or the ratio of coverage.

shown in Figure 5. The improvement on GDumb is modest at low noise rate but noticeable at high noise rate especially on TACRED. It indicates that although the quality of samples in the replay buffer is poor, the online learning process leads to better relational representations, and eventually outperforms the original dumb learner by noticeable margins. Note that the performance of GS + OL on TACRED with 80% noise is close to S6. We speculate that the advantage of sample separator is slight under this noisy and imbalanced situation because its ability to select clean samples into the clean replay buffer is similar to the greedy sampler, which also demonstrates the effectiveness of online learning.

Importance of Noisy Memory

In semi-supervised offline finetuning, the learner utilizes both the clean and noisy replay buffers. To show the necessity of utilizing noisy samples, we conduct experiments on FewRel and TACRED both with 80% noise and the NYT-H. The results are illustrated in Tabel 3. Utilizing both noisy and clean samples consistently outperforms only utilizing clean samples or utilizing twice clean samples. There are three reasons for this phenomenon. First, there is a lot of noise in both datasets (80% and 64.63%). It is very difficult to maintain a good clean replay buffer, and will inevitably bring a lot of noise into the buffer. Therefore, even if the buffer capacity is expanded, more noisy rather than clean samples are stored in replay buffer, resulting in little improvement. Second, the samples in clean replay buffers may have close MSE losses and tend to be similar, while utilizing the noisy samples enhances data diversity. Third, as mentioned before, sometimes the relations in the clean replay buffer cannot cover the whole relation set in extremely imbalanced dataset such as NYT-H. If we just train the learner on the clean replay buffer, the learner has no chance to access these rare relation information and would degrade the final performance. Although without the supervised signals, incorporating their unsupervised features in a semi-supervised fashion is beneficial for finetuning. This method could also serve as a remedial measure to ensure the participation of all the relations. More powerful purifying and sampling methods are worth exploring in the future.

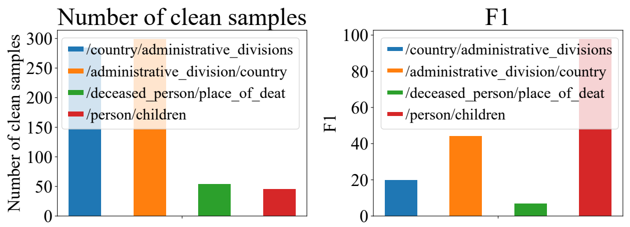

Figure 6: Case study on NYT-H.

Difficulties of Handling Real-world Noise

To provide a deeper understanding of the difficulties of handling real-world noise, we analyze two cases shown in Figure 6. We find the imbalance of samples belonging to relations with similar semantics and the strong assumption of distant supervision make some relations hard to learn reasonably. Here we have two pairs of relations that both have similar number of clean samples. The relations */location/country/administrative_divisions* and */location/administrative_division/country* are symmetric and tend to have similar difficulties for models to learn. But the performance of the relation */location/country/administrative_divisions* is clearly poor because the relation */location/location/contains* is the most frequently appearing relation in NYT-H. Obviously, these two relations have very similar semantics (the head entity is a country and the tail entity is a city in most cases), so the learning attention of models is mostly attracted by this common relation. Moreover, the performance gap between */people/deceased_person/place_of_death* and */people/person/children* is even greater. We examine the relation set in NYT-H and find two other relations (*/people/person/place_lived* and */people/person/place_of_birth*) that easily cause the noise under distant supervision because it is common that the place of birth, live and death for a person remain the same in real-world, which means many entity pairs are labeled with three relations simultaneously. So the models become confusing when trying to understand these three relations. We notice the performance on relation */people/person/children* is well for two reasons. On the one hand, different from */people/deceased_person/place_of_death*, there is no other relation to interfere with it. On the other hand, its noise rate is only around 20%, which is not so difficult for online purifying and learning.

Conclusion

This paper defines a more difficult but realistic continual relation learning setting called ON-CRE that requires models to address both online and noisy relation learning scenarios. We propose a novel continual relation learning framework called S6 consisting of sample separated online purifying, self-supervised online learning and semi-supervised offline finetuning. Experimental results show that S6 outperforms the state-of-the-art models substantially in ON-CRE, providing a good solution and prospect for more realistic CRE. We also analyze the limitation of the purifying technique and the difficulties of handling real-world noise, encouraging more effective methods in the future research.

Acknowledgements

We thank the anonymous reviewers for their insightful comments. The work was supported in part by the 13th Five-Year All-Army Common Information System Equipment Pre-Research Project (Grant Nos. 31514020501, 31514020503). All opinions are of the authors and do not reflect the view of sponsors.

References

Aljundi, R.; Belilovsky, E.; Tuytelaars, T.; Charlin, L.; Caccia, M.; Lin, M.; and Page-Caccia, L. 2019. Online Continual Learning with Maximal Interfered Retrieval. In *NeurIPS*.

Arpit, D.; Jastrzkebski, S.; Ballas, N.; Krueger, D.; Bengio, E.; Kanwal, M. S.; Maharaj, T.; Fischer, A.; Courville, A.; Bengio, Y.; et al. 2017. A closer look at memorization in deep networks. In *ICML*.

Baldini Soares, L.; FitzGerald, N.; Ling, J.; and Kwiatkowski, T. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *ACL*.

Chen, T.; Goodfellow, I. J.; and Shlens, J. 2016. Net2Net: Accelerating Learning via Knowledge Transfer. In *ICLR*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*.

Cui, L.; Yang, D.; Yu, J.; Hu, C.; Cheng, J.; Yi, J.; and Xiao, Y. 2021. Refining Sample Embeddings with Relation Prototypes to Enhance Continual Relation Extraction. In *ACL*.

de Masson d'Autume, C.; Ruder, S.; Kong, L.; and Yogatama, D. 2019. Episodic Memory in Lifelong Language Learning. In *NeurIPS*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

Grandvalet, Y.; and Bengio, Y. 2004. Semi-supervised Learning by Entropy Minimization. In *NeurIPS*.

Han, X.; Dai, Y.; Gao, T.; Lin, Y.; Liu, Z.; Li, P.; Sun, M.; and Zhou, J. 2020. Continual Relation Learning via Episodic Memory Activation and Reconsolidation. In *ACL*.

Han, X.; Zhu, H.; Yu, P.; Wang, Z.; Yao, Y.; Liu, Z.; and Sun, M. 2018. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In *EMNLP*.

Karim, N.; Khalid, U.; Esmaeili, A.; and Rahnavard, N. 2022. CNLL: A Semi-Supervised Approach for Continual Noisy Label Learning. In *CVPR*.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2020. Supervised Contrastive Learning. In *NeurIPS*.

Kim, C. D.; Jeong, J.; Moon, S.; and Kim, G. 2021. Continual Learning on Noisy Data Streams via Self-Purified Replay. In *ICCV*.

Kun, Y.; and Jianxin, W. 2019. Probabilistic End-to-end Noise Correction for Learning with Noisy Labels. In *CVPR*.

Li, G.; Chen, X.; Wang, P.; Xie, J.; and Luo, Q. 2022. FastRE: Towards Fast Relation Extraction with Convolutional Encoder and Improved Cascade Binary Tagging Framework. In *IJCAI*.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Miyato, T.; Dai, A. M.; and Goodfellow, I. 2017. Adversarial training methods for semi-supervised text classification. In *ICLR*.

Peng, M.; Liao, Q.; Hu, W.; Tian, G.; Wang, H.; and Zhang, Y. 2019. Pattern Filtering Attention for Distant Supervised Relation Extraction via Online Clustering. In *WISE*.

Pleiss, G.; Zhang, T.; Elenberg, E.; and Weinberger, K. Q. 2020. Identifying Mislabeled Data using the Area Under the Margin Ranking. In *NeurIPS*.

Prabhu, A.; Torr, P.; and Dokania, P. 2020. GDumb: A Simple Approach that Questions Our Progress in Continual Learning. In *ECCV*.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *CVPR*.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience Replay for Continual Learning. In *NeurIPS*.

Wang, H.; Xiong, W.; Yu, M.; Guo, X.; Chang, S.; and Wang, W. Y. 2019a. Sentence Embedding Alignment for Lifelong Relation Extraction. In *NAACL*.

Wang, Y.; Chen, M.; Zhou, W.; Cai, Y.; Liang, Y.; and Hooi, B. 2022. GRAPHCACHE: Message Passing as Caching for Sentence-Level Relation Extraction. In *Findings of NAACL*.

Wang, Y.; Ma, X.; Chen, Z.; Luo, Y.; Yi, J.; and Bailey, J. 2019b. Symmetric cross entropy for robust learning with noisy labels. In *ICCV*.

Wei, H.; Feng, L.; Chen, X.; and An, B. 2020. Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization. In *CVPR*.

Wei, J.; and Zou, K. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *EMNLP*.

Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; and Zhao, J. 2014. Relation Classification via Convolutional Deep Neural Network. In *COLING*.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual Learning Through Synaptic Intelligence. In *ICML*.

Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; and Manning, C. D. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *EMNLP*.

Zhao, K.; Xu, H.; Yang, J.; and Gao, K. 2022. Consistent Representation Learning for Continual Relation Extraction. In *Findings of ACL*.

Zhu, T.; Wang, H.; Yu, J.; Zhou, X.; Chen, W.; Zhang, W.; and Zhang, M. 2020. Towards Accurate and Consistent Evaluation: A Dataset for Distantly-Supervised Relation Extraction. In *COLING*.