# The replication and improvements of Efficient Parameter Isolation method

He Hangjun

10th, December 2023

Abstract

To address the issue of Catastrophic forgetting of Continual Learning, the method shown in this paper uses a type of Parameter Isolation method to separate the conflicted learning processes of tasks arrived at different times. The paper facilitates Pretrained Language Model to obtain task specific parameters, and devises a new technique to choose right parameters for newly arrived tasks. Furthermore, it also uses partial weights masking to reduce storage costs and enhance privacy. This method performs better than all other no replay methods, and even outperforms some historical data cache methods. Based on the original paper, we devise a new method to transfer knowledge. After that, We establish the rationale of transferring knowledge by initialization.

Keywords: Continual Learning, Catastrophic Forgetting, Parameter Isolation.

## 1 Introduction

Continual Learning (CL) is about tackling the problem that learning tasks are not arriving all at once. It needs to deal with dynamic tasks' distribution instead of commonly static settings. And it can fit well in the context of adaptive learning as the tasks are constantly changing their forms in the real world. However, CL is known for its infamous issue – Catastrophic Forgetting (CF). CF is a phenomenon that models perform worse in previous tasks after training new task. There are several ways to mitigate this problem. The paper advocates a new method called Efficient Parameter Isolation to address issue of CF. This method makes great progress comparing to other related methods.

## 2 Related works

### 2.1 Rehearsal-based methods

By storing the previously seen training samples, and replaying them [1] when new task arrives, CF can be alleviated to some extent. But it requires highly storage costs. To further reduce such costs, compressing the old data for future training is another feasible solution[2]. With introduction of compressing, it is natural to use Knowledge Distillation to lower the complexity of old data[3]. Despite all of those improvements, the expense of preserving is still no trivial.

### 2.2 Regulation-based methods

This category of methods mixes old and new tasks in different weights with carefully chosen restriction terms[4], and they need to maintain a copy of old model which is not storage effective.

With facilitating Knowledge Distillation and conducting the reduction of unrelated weights and enhancement of previously trained tasks' weight of parameters, the knowledge can be preserved[5] through subsequent training process. However, the generalization ability of model is reduced.

### 2.3 Parameter Isolation methods

For isolating parameters between tasks, those methods allocate task specific parameters. The limitation of them is requiring specific task identity which prevent them from further extrapolating. The described method in the paper belongs to this category but has some improvements comparing to previous works.

## 3 Method

### 3.1 Overview

The outline of proposed method is as shown in Figure 1:

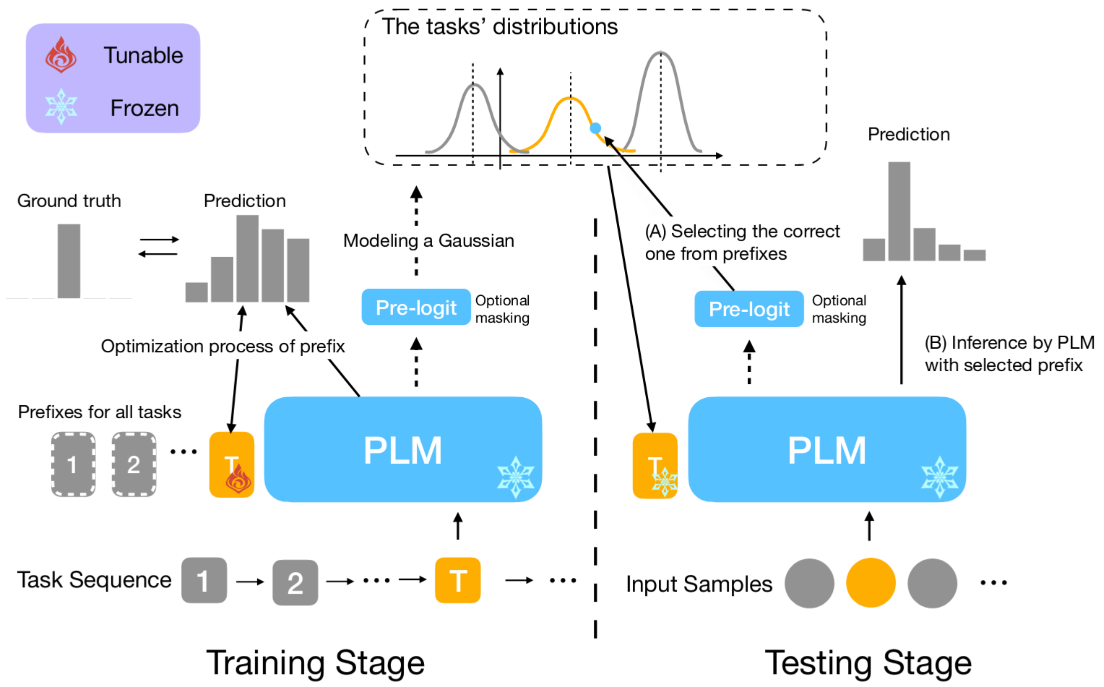

Figure 1. Overview of the method

On the training stage, for each task, the method adds virtual tokens as delta parameters[6] to the beginning of tasks' input, and feeds input to every Multi-head Attention layer of Pretrained Language Model(PLM). The weights of PLM are frozen at both training and testing stages. On the testing stage, when a new task arrives, the method initializes new task's prefix according to previously trained tasks' prefix. To choose which task specific parameters to apply, the paper computes statistics of all previously learned tasks' prefixes and compares them with current task's prefix through distance metric. Note that on the testing stage, the task-specific parameters are frozen, and the saved mask is read to yield distance.

### 3.2 Task identification

After new task arrived, we need a way to determine which task-specific parameters should be applied on it. To achieve that, the paper uses Gaussian Discriminant analysis to compute the prelogits of task's through PLM. Then, it uses the prelogits to yield means in Equation 1 and covariance matrices in Equation 2. At next step, the paper calculates the Mahalanobis distance between task's prefix and distribution of previously trained tasks and chooses the nearest task's prefix. The Mahalanobis Distance is defined as Equation 3.

$$\mu_t^c = \frac{1}{N_c} \sum_{y_i=c} h(x_i) \tag{1}$$

$$\Sigma_t = \frac{1}{N} \sum_{c} \sum_{y_i=c} (h(x_i) - \mu_t^c)(h(x_i) - \mu_t^c)^T \tag{2}$$

$$d(x, c) = -(h(x) - \mu_t^c)^T (\Sigma_t)^{-1} (h(x) - \mu_t^c) \tag{3}$$

### 3.3 Knowledge transfer

The parameter isolation method separates the conflicted parameters, but it also stops new task from taking advantage of previously trained tasks' knowledge, which renders inefficient training process. To solve this problem, the paper introduces two knowledge sharing methods. One is prefix fusion, which prepends all previously trained tasks' prefix to current task's prefix and feeds them into the PLM to generate new prefix. The other is initializing prefix from previous tasks' prefix. Specifically, current task's prefix will be assigned with mean of previous tasks' prefix, as $Prefix_{current} = \frac{1}{n}\Sigma Prefix_i$, or last trained prefix.

## 4 Implementation details

### 4.1 Comparing with the released source codes

The source code of this paper is publicly available. Our work is mainly based on existing code. Beyond that, we enhance the proposed method by devising a new knowledge transfer method.

### 4.2 Datasets

We use the 5ds[7] dataset provided by the paper, and the WOS[8] dataset available on huggingface.co.

### 4.3 Experimental environment setup

Our experimental environment setup is listed on Table 1.

| | |
|---|---|
| Graphic Card | NVIDIA GeForce RTX 4090 |
| Processor | Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60GHz |
| Operating System | Ubuntu 20.04 |
| Container | Docker version 24.0.2 |
| Python Version | 3.7 |
| Python Libraries | transformers 4.23.1, datasets 2.1.0, pytorch 1.11.0 |

Table 1. Experimental environment

### 4.4 Interface design

We can run code with command listed below to evaluate method. The 'prompt_fusion_mode' argument denotes which knowledge transfer method to use. The 'n_per_class' argument denotes shot for every class.

```
python run.py \
    --dataset {5ds|WOS} \
    --prompt_fusion_mode {None|fusion|last|mean|SMD} \
    --method epi \
    --query_mode mahalanobis \
    --prompt_mode prefix \
    --pre_seq_len 32 \
    --n_per_class {2000|10|20|50} \
    --batch_size 32 \
    --lr 0.05 \
    --epochs 5
```

### 4.5 Main contributions

#### 4.5.1 Another knowledge transfer method

We devise a new knowledge transfer based on paper's methods – sample mahalanobis distance(SMD). The intuitive behind this method is that the smaller distance between sample and task's distribution is, the higher chance the new task's distribution can converge in shorter time. This method initializes the new prefix using previous parameters having the lowest scores in mahalanobis distance(MD) and the averaged prefix with the factor $\alpha$.

First, we use Equation 3 to compute mahalanobis distances between the sample and each task's trained prefix over classes as score vectors. And we calculate the mean $m$ and variance $v$ for every score vector. We obtain the scale factor $\theta$ from maximum of means $\mathbf{m}$ and maximum of variances $\mathbf{v}$ as shown in Equation 4.

$$\theta = \frac{\max \mathbf{m}}{\max \mathbf{v}} \tag{4}$$

After that, we compute $\mu_k$ the inverse of the weighted sum of mean and variance as shown in Equation 5.

$$\mu_k = \frac{1}{m_k + \theta v_k} \tag{5}$$

Then, we get top k elements of array $\mu$ as below. We denote the number of tasks as $n$. The value of $k$ is half of the number of tasks.

$$K = \{k|k \in topk([\mu_i, i = 1...n])\} \tag{6}$$

From top k parameters, we get the weighted sum prefix as Equation 7.

$$P_m = \sum_{k \in K} \frac{\mu_k}{\sum_i \mu_i} P_k \tag{7}$$

To smooth the gap between the top k closest prefixes' sum $P_m$ and global mean, we use the factor $\alpha$ based on the number of shot $s$ to determine weight of terms. The definition of $\alpha$ is illustrated in Equation 8.

$$\alpha = \frac{1}{\log_2 s} \tag{8}$$

Finally, we reach Equation 9 to compute current prefix $P_c$ where $\frac{1}{n}\sum_{i=1}^{n} P_i$ is the mean of all prefixes.

$$P_c = \alpha P_m + (1 - \alpha)\frac{1}{n}\sum_{i=1}^{n} P_i \tag{9}$$

#### 4.5.2 Rationale of transferring knowledge by initialization

Since the original paper did not explain the reason of using initialization to transfer knowledge, we give a rationale here.

We discuss both cases of convex and non-convex loss functions.

In the case of convex loss functions, functions should converge to a global minimum after finite steps in certain learning rate. So the similar delta parameters will be able to decrease steps toward convergence, and virtually accelerate training process.

Now considering the scenario of non-convex functions, functions maybe converge to a local minimum depending on what the initial weights are. Thus, to avoid being trapped around a local minimum, a proper initial value is needed.

Both cases display the soundness of transferring knowledge by initialization.

## 5 Results and analysis

As shown in Table 2, results of our replication approaches the paper's in 2000 shot setting. The result is consistent with the paper's outcome.

| Method | 5ds | WOS |
|---|---|---|
| EPI | 74.43 | 77.83 |
| EPI(Ours) | 74.34 | 77.84 |

Table 2. Accuracy in 2000 shot setting

Since tasks in 5ds datasets don't have many similarities, knowledge transfer will not perform well. We only conduct knowledge transfer on WOS datasets. As shown in Table 3, our method surpasses all four methods in 10-shot and 50-shot settings, while it is not good as Mean and Last methods in 20-shot setting.

| Method | 10-shot | 20-shot | 50-shot |
|---|---|---|---|
| None | 34.71 | 46.43 | 58.06 |
| Fusion | 37.01 | 47.69 | 60.55 |
| Mean | 33.40 | 49.27 | 61.79 |
| Last | 33.44 | 50.89 | 61.70 |
| SMD(Ours) | 39.55 | 49.07 | 62.04 |

Table 3. Accuracy in different shots and methods on WOS

## 6 Conclusion and future work

Efficient Parameter Isolation, as a parameter isolation method, is able to utilize PET and task identification to effectively mitigate CF in Continual Learning. But the mechanism of parameter isolation method poses a challenge to transfer knowledge to other tasks. Based on knowledge transfer methods of this paper, we implement a new method called SMD and obtain higher results than most of the methods in the measurement of accuracy. However, currently no such knowledge transfer method can perform well on all datasets, thus a more robust knowledge transfer method is needed.

## References

[1] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[2] Lucas Caccia, Eugene Belilovsky, Massimo Caccia, and Joelle Pineau. Online learned continual compression with adaptive quantization modules. In International conference on machine learning, pages 1240–1250. PMLR, 2020.

[3] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 2001–2010, 2017.

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

[5] Zhizhong Li and Derek Hoiem. Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence, 40(12):2935–2947, 2017.

[6] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pages 2790–2799. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

[7] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. Advances in neural information processing systems, 28, 2015.

[8] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In 2017 16th IEEE international conference on machine learning and applications (ICMLA), pages 364–371. IEEE, 2017.