

**VIETNAM NATIONAL UNIVERSITY,
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**



Pham Thi Quynh Trang

**MITIGATING CATASTROPHIC FORGETTING IN CONTINUAL LEARNING
VIA MEMORY REPLAY, KNOWLEDGE TRANSFER,
AND ADAPTIVE ARCHITECTURE OPTIMIZATION**

**(GIẢM THIỂU QUÊN NGHIÊM TRỌNG TRONG HỌC LIÊN TỤC SỬ DỤNG
CÁC PHƯƠNG PHÁP PHÁT LẠI BỘ NHỚ, CHUYỂN GIAO TRI THỨC
VÀ KIẾN TRÚC ĐỘNG)**

LIST OF PUBLICATIONS

Major: Information Systems

Code: 9480104

Hanoi - 2026

**VIETNAM NATIONAL UNIVERSITY,
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

Pham Thi Quynh Trang

**MITIGATING CATASTROPHIC FORGETTING IN CONTINUAL LEARNING
VIA MEMORY REPLAY, KNOWLEDGE TRANSFER,
AND ADAPTIVE ARCHITECTURE OPTIMIZATION**

**(GIẢM THIỂU QUÊN NGHIÊM TRỌNG TRONG HỌC LIÊN TỤC SỬ DỤNG
CÁC PHƯƠNG PHÁP PHÁT LẠI BỘ NHỚ, CHUYỂN GIAO TRI THỨC
VÀ KIẾN TRÚC ĐỘNG)**

LIST OF PUBLICATIONS

Major: Information Systems

Code: 9480104

Supervisor: Assoc. Prof. Nguyen Tri Thanh

Co-supervisor: Dr. Le Duc Trong

Hanoi - 2026

List of Publications

- [P1] **Quynh-Trang Pham Thi**, Anh-Cuong Pham, Ngoc-Huyen Ngo and Duc-Trong Le. "Memory-Based Method using Prototype Augmentation for Continual Relation Extraction." In 2022 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1-6. IEEE, 2022. Scopus.
- [P2] Thanh-Hai Dang, **Quynh-Trang Pham Thi**, Duc-Trong Le, and Tri-Thanh Nguyen. "Contrastive Learning for Boosting Knowledge Transfer in Task-Incremental Continual Learning of Aspect Sentiment Classification Tasks". VNU Journal of Science: Computer Science and Communication Engineering 40, no. 1 (2024).
- [P3] Duc-Hung Nguyen, Tri-Thanh Nguyen, Thanh-Hai Dang and **Quynh-Trang Pham Thi**. "ZeroRFF: A Random Fourier Features and Analytic Learning Method for Generalized Class Incremental Learning". The 21st International Conference on Advanced Data Mining and Applications 2025. Scopus Rank B.
- [P4] **Quynh-Trang Pham Thi**, Duc-Hung Nguyen, Thanh Hai Dang, Duc-Trong Le, Tri-Thanh Nguyen, and Quang-Thuy Ha. "Towards Robust Continual Learning: A Multi-Head Approach with Online Prototype Equilibrium and Adaptive Prototypical Feedback." In Asian Conference on Intelligent Information and Database Systems, pp. 275-286. Singapore: Springer Nature Singapore, 2024. Scopus Rank B.
- [P5] **Quynh-Trang Pham Thi**, Duc-Hung Nguyen, Anh-Duc Nguyen-Tran, Hung-Dung Nguyen, Tri-Thanh Nguyen, Quang-Thuy Ha and Thanh Hai Dang. "Addressing Catastrophic Forgetting in Continual Learning using Multihead Approach and Latent Alignment via Energy-Based Model." Vietnam journal of computer science. (Accepted). WoS Q3.
- [P6] **Quynh-Trang Pham Thi**, Van-Truong Le, Thanh-Huyen Pham, Thanh-Nga Hoang Thi and Duc-Trong Le. "DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic in Continual Learning". SOICT 2025. Scopus.

[P7] **Quynh-Trang Pham Thi**, Quang-Dat Mac, Thanh-Ha Nguyen and Duc-Trong Le
."SO-LoRA: Sparse Orthogonal LoRA for Parameter-Efficient Continual Learning".
In 2026 IEEE International Conference on Multimedia and Expo (ICME 2026).
Scopus Rank A.

The list contains seven publications.

Memory-Based Method using Prototype Augmentation for Continual Relation Extraction

1st Quynh-Trang Pham Thi

Vietnam National University
University of Engineering and Technology
Hanoi, Vietnam
trangptq@vnu.edu.vn

2nd Anh-Cuong Pham

Vietnam National University
University of Engineering and Technology
Hanoi, Vietnam
19020038@vnu.edu.vn

3rd Ngoc-Huyen Ngo

Vietnam National University
University of Engineering and Technology
Hanoi, Vietnam
19020047@vnu.edu.vn

4th Duc-Trong Le

Vietnam National University
University of Engineering and Technology
Hanoi, Vietnam
trongld@vnu.edu.vn

Abstract—Continual relation extraction is a field that applies continual learning in relation extraction, which aims to train the model with new relations without losing the accurate classification of old ones. There are two significant challenges in continual Learning: catastrophic forgetting and knowledge transfer. Some previous work has shown that the memory-based method, which stores a few training examples in old tasks and retrains them while training new tasks, can solve these problems of continual learning, and it is well-performed in natural language processing tasks. However, memory-based methods tend to be overfitting and low-performed on imbalanced datasets. A combination of consistent representation learning and the prototype augmentation mechanism is proposed to solve this problem. We also reduce the real data storage by using the virtual data points, instead of using all data are the real ones. These virtual data points are generated by the prototypes of relations obtained through the data augmentation process. Experiments on FewRel and TACRED datasets reveal that our method performs better than the latest baselines in catastrophic forgetting.

Index Terms—continual learning, continual relation extraction, prototype augmentation

I. INTRODUCTION

Relation Extraction (RE) is a task where attributes and relations of entities are extracted from a sentence. It is a subtask of Information Extraction (IE) that plays an essential role in Natural Language Processing (NLP) applications such as structured search, sentiment analysis, question answering, and summarization [1]. However, the conventional RE models always focus on working with predefined relations, which cannot handle the rapid emergence of new relations in real life [2].

In the process of finding solutions for this problem, Wang et al. proposed an embedding approach

This work has been supported by VNU University of Engineering and Technology under project number CN22.07.

to enable continual learning for relation extraction tasks and opened up Continual Relation Extraction (CRE) - a new direction for the study of relation extraction problem [3]. However, continual learning has a notable drawback, which is catastrophic forgetting. Catastrophic forgetting is the phenomenon that happens when the model forgets the previous knowledge after learning the new ones. And the aim of CRE is to help the model learn new relations without catastrophic forgetting [4].

Recent studies have proposed solutions for alleviating catastrophic forgetting in continual Learning, mainly concentrated on regularization methods [5], [6], dynamic architecture methods [7], [8], and memory-based methods [4], [9], [10]. Memory-based approaches, which maintain memory to save some training examples in old tasks and retrain memorized examples at the same time when training new tasks, are shown to have the best performance to the catastrophic forgetting problem, especially in NLP tasks [3], [4], [11]. However, memory-based methods have trouble overfitting with memory samples and are low-performed with imbalanced datasets [4]. A contrastive representation learning method is proposed by Zhao et al. to solve these challenges by using supervised contrastive learning and knowledge distillation to restrict old tasks with no dramatic change [4].

One another remarkable memory-based method is the Prototype Augmentation (protoAug) mechanism, which is proposed in the study by Zhu et al. [12]. Prototype Augmentation is intended to maintain the distinction and balance between old and new classes, and prevent significant change in the decision boundary, thus solving the challenges of catastrophic forgetting [12]. In fact, storing data from previous tasks is limited by memory or privacy issues, and protoAug

mechanism can be used to deal with this issue.

In this study, we expand the work of Zhao et al. about CRL to take advantage of stability and adopt prototype augmentation in this idea to improve the ability to limit the catastrophic forgetting of CRL [4]. Specifically, the combined method we propose uses an alternative approach that limits the real data stored from the tasks before. We make utilize of a data augmentation mechanism to generate virtual data points based on the prototype of previous relations in the training process. With this approach, this model can minimize real data usage, thereby minimizing the issue of limitations in data storage. We conduct experiments with two RE datasets: FewRel and TACRED, and the results show that our proposed model outperforms the state-of-the-art baselines in alleviating catastrophic forgetting for CRE. In conclusion, our main contributions are summarized as follows:

- We propose a method that uses the prototype of each relation to generating pseudo data instead of adding all of the real data into the memory bank for training in the later task, which helps the model alleviate catastrophic forgetting.
- We evaluate our proposed model on two RE datasets: FewRel and TACRED, and show that it performs stable through training, which means it is better than the state-of-the-art baselines in alleviating catastrophic forgetting.

II. RELATED WORK

A. Continual Learning

Traditional machine learning aims to learn single tasks, so this method cannot reuse knowledge that models learned in the past. Continual Learning tries to learn like humans and accumulate knowledge not only to solve new problems but also to improve the performance of old tasks with new knowledge. There are two main challenges in Continual Learning: Catastrophic Forgetting and Transfer Knowledge. Catastrophic Forgetting happened when the model tries to update weights to optimize performance on new tasks leading to the results of old tasks declining significantly.

There are some main methods for dealing with these two problems : (1) Regularization-based methods impose constraints on updating neural weights important to previous tasks for relieving catastrophic forgetting. (2) Dynamic architecture methods extend the model architecture dynamically to learn new tasks and prevent forgetting old tasks effectively. (3) Memory-based methods save some samples from old tasks and continuously learn them in new tasks to alleviate catastrophic forgetting [4]. Inspired by the success of the memory-based method [3], [11], [14], we studied and applied this approach for the continual relation extraction tasks.

B. Contrastive Learning

Contrastive Learning (CL) aims to make the representations of similar samples mapped closer, and

dissimilar samples mapped farther in the embedded space [15]. In recent years, contrastive learning has been raised, and this method is promising when applied to self-supervised representation learning. One notable research at this time is supervised contrastive learning, proposed by Khosla et al., which gained much attention [16]. In 2022, inspired by the research on supervised contrastive learning, Zhao et al. proposed a better representation of Learning by applying contrastive learning [4]. In this paper, we used contrastive representation learning to get great representations for classes while training.

III. METHODOLOGY

In this section, we introduce our method in detail.

A. Task Formalization

Continual Relation Extraction trained a model in a sequence of tasks $\{T_1, T_2, \dots, T_K\}$ with K as the number of tasks. Each k -task T_k has its own training set D_k and relation set R_k . The T_k task is a supervised classification task, including a sequence of examples and their corresponding labels $\{x_i, y_i\}_{i=1}^N$, with x_i is the input data, including natural language text and entity pair, $y_i \in R_k$ is the relation label.

The goal of this problem is to train the model so that it can learn from new tasks without forgetting knowledge from the previous ones. Specifically, the model will be trained on D_k at the k -th step to learn new relations in R_k . Since the relations will be generated and accumulated, the continuous relation extraction problem requires the models to perform well at both the k -th task and $k - 1$ previous tasks. The model, after training on T_k will be able to classify all known relations on the composite set of relations $\hat{R}_k = \bigcup_{i=1}^k R_i$.

Inspired by previous research about CRE models by Zhao et al. [4], we use an episodic memory module to store typical samples of each relation that the model has learned in old tasks. The memory module for relation r is represented as a memorized sample set $M_r = \{(x_1, r), \dots (x_O, r)\}$, where each sample is labeled with r , and O is the memory size (sample number). Therefore, the episodic memory for the observed relations in $T_1 \sim T_k$ is $\hat{M}_k = \bigcup_{r \in \hat{R}_k} M_r$.

B. Model Pipeline

The learning procedure of our model for a current task T_k contains four major steps:

Initial Training Encodes the input via the encoder **E**. Then, the parameters from the encoder **E** and the projector *Proj* are tuned with the current training samples in D_k using supervised contrastive learning.

Sample selection For each relation $r \in R_k$ that was unobserved in the previous tasks, we retrieve all samples with label r in D_k . Then, we use the k-means algorithm to cluster these samples. For each cluster, the sample closest to the centroid is chosen to represent that cluster. Let's call it M_r . After that,

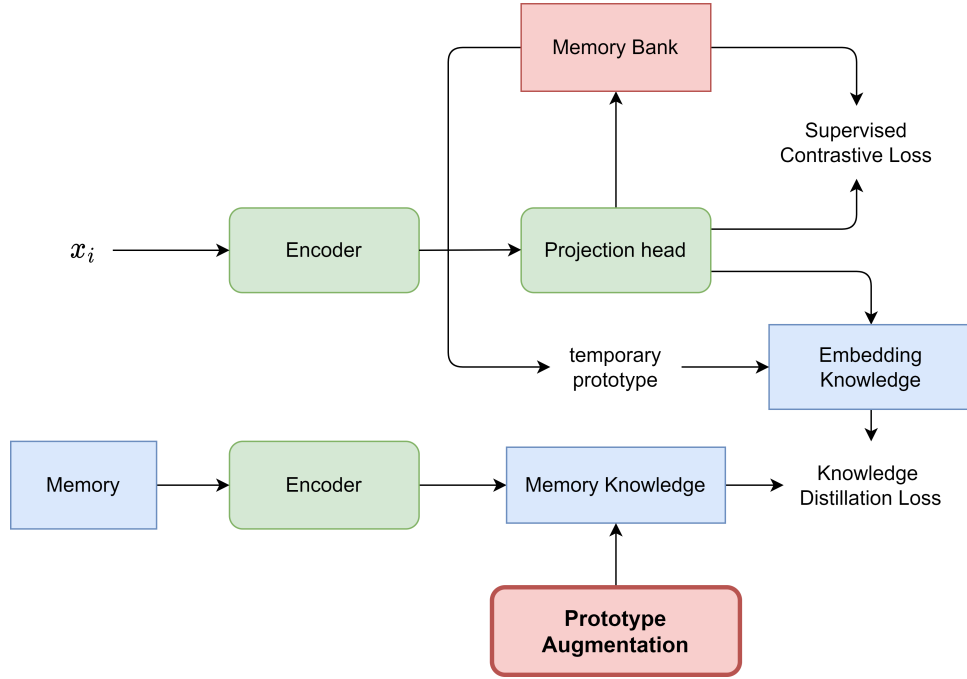


Fig. 1. Module of consistent representation using prototype augmentation

generate prototype p_r of r based on M_r to expand the prototype set $\mathbf{P}_r$.

Memory augmentation based on prototype Data storing from previous tasks might be limited because of memory or privacy issues. Therefore, in this part, we proposed a simple non-exemplar-based method, which uses prototype-based memory augmentation to solve the catastrophic forgetting problem in continual learning. On the one hand, protoAug memorizes a one-class representation prototype for each old class and augments memorized archetypes via Gaussian noise when learning new classes. Then, augmented prototypes and memory stored are joined to maintain the discrimination and balance between old and new classes.

Memory replay After the augmentation phase, combine the newly generated pseudo data and data in memory. And then train them as a bigger memory. In this process, memory replay is also applied iteratively to learn new relation prototypes while strengthening the discriminability of old relational prototypes.

C. Sample Encoder

Given an example x , we adopt an encoder to encode its semantic features for detection. To be specific, we first tokenized the given example into several tokens and then input the tokenized tokens into neural networks to compute their corresponding embedding. As extracting relations from sentences is related to those entities mentioned in sentences, we thus add special tokens into the tokenized tokens to indicate the beginning and ending positions of those entities.

Specifically, we use $[E_{11}]$, $[E_{12}]$, $[E_{21}]$ and $[E_{22}]$ to denote the start and end position of head and tail entity, respectively. Next, a sample's hidden representation is

the concatenation of token embeddings of $[E_{11}]$ and $[E_{21}]$, which has been proven effective in previous works [20].

$$h = W[h_{[E_{11}]}; h_{[E_{21}]}] + b$$

where $h_{[E_{11}]}, h_{[E_{21}]} \in \mathcal{R}^{d_h}$ (d_h is the dimension of BERT hidden representation) are the hidden representations of $[E_{11}]$ and $[E_{21}]$, $W \in \mathcal{R}^{2d_h \times d_h}$ (d is sample embedding dimension) and $b \in \mathcal{R}^{d_h}$ are trainable parameters. The encoder is denoted as $\mathbf{E}$. Then, we use a projection head $Proj$ to obtain the low-dimensional embedding: For down dimensional of embedding, we feed output Encoder $\mathbf{E}$ to projector $Proj$:

$$\tilde{z} = \text{MLP}(h)$$

With MLP is Multi-layer Perception. Finally, embedding $z = \tilde{z}/\|\tilde{z}\|$ is used for contrastive learning, and the hidden representation is used for classification.

D. Initial Training for New Task

According to the general assumption of CRE, all relations in R_t are unobserved in former tasks $T_1 \sim T_{k-1}$. We first introduce the model's initial training on the current task for a new task. After feeding each batch B into encoder $\mathbf{E}$, projector $Proj$ and normalized output, we use supervised contrastive loss [16] to pull all samples in the same class closer and push others far away:

$$\mathcal{L}_{CL} = \sum_{i=1}^{|B|} \frac{-1}{N_{y_i}} \sum_{j=1|y_i=y_j}^{|B|} \log \frac{\exp(z_i \cdot z_j / \tau)}{\sum_{b=1}^{|B|} \exp(z_i \cdot z_b / \tau)}$$

Where N_{y_i} is the number of samples whose output is y_i . $\tau \in \mathcal{R}^+$ is a tunable parameter used to control

the division of layers. The symbol $\cdot$ is used to represent the dot product.

E. Sample selection

This paper focuses on how to use limited memory resources effectively. Before training the model, we will determine the total memory size M according to the limited memory resources and the scale of tasks. The number of stored samples for each task depends on the total memory size M and learned task number t . Following the principle of making the best use of storage space, the memory size for task T_k is:

$$M_k = \min(\max(M_{left}, \frac{M}{k}), |D_k|)$$

Where M_{left} is the available memory left, D_k is the number of training samples in current task training data. To ensure the model performs well on all tasks, memory M is averagely allocated for all seen tasks. The samples newly selected will replace the old samples when the memory is full. First, we find the tasks in which M has memory space M_i bigger than average space $\frac{M}{k}$. Then $M_i - \frac{M}{k}$ samples will be randomly selected to be replaced until there is enough space M_k for the new task. This strategy can ensure that the number of samples for each task in the memory is balanced, which can effectively alleviate overfitting and catastrophic forgetting.

When selecting samples from the training data D_k for storage, random selection is not an ideal way. The samples selected randomly cannot well cover the distribution of relations. To select more representative samples in the task, k-means clustering is used here. The embedding features of the text in training data are clustered into M_k classes using k-means. The sample closest to the center of each class is selected to be stored in the memory M , which contains more information and better covers the feature distribution of tasks.

F. Memory augmentation based on prototype

Storing data from previous tasks is limited by memory or privacy issues, and generative models are usually unstable and inefficient in training. In this paper, we propose augmentation memory based on data distribution to mitigate catastrophic forgetting problems.

We compute the expectation feature of each class using memory stored, for class c in task k we get:

$$\mu_{k,c} = \frac{1}{|M_c^k|} \cdot \sum_{i=1}^{|M_c^k|} \mathbf{E}(x_i)$$

There for the feature generated for class c in task k is augmented below:

$$f_{k,c} = \mu_{k,c} + \epsilon * r$$

Where $\epsilon \sim \mathcal{N}(0,1)$ is the derived Gaussian noise which has the same dimension as a prototype. r is a scale to control the uncertainty of the augmented

prototypes. In particular, the scale r can be predefined or computed as the average variance of the class representations:

$$r_{k,c}^2 = \sum_{i=1}^{|M_{k,c}|} \frac{\text{Tr}(\sum_{k,c})}{D}$$

Where D is the dimension of the deep feature space. $\sum_{k,c}$ is the covariance matrix for the features from class c at stage k , and the Tr operation computes the trace of a matrix. After the memory is augmented, we join this with our memory and train the next step, memory relay.

G. Memory Replay

After the initial training for the current task T_k , old relations' embedding space is likely to be disrupted because the model is tuned towards fitting T_k 's learning objective. Inspired by [12], two replay strategies to learn consistent representation for alleviating this problem: contrastive replay and knowledge distillation on memory. One difference from the previous study is we augmented memory to alleviate catastrophic forgetting.

First, we use contrastive Learning for replay memory. This step is similar step initial training for a new task; the difference here is that each batch uses all the samples in the entire memory bank for training:

$$\mathcal{L}_{CE} = \sum_{i=1}^{|M'_k|} \frac{-1}{N_{y_i}} \sum_{j=1|y_i=y_j}^{|M'_k|} \log \frac{\exp(z_i \cdot z_j / \tau)}{\sum_{m=1}^{|M'_k|} \exp(z_i \cdot z_m / \tau)}$$

Where M'_k is memory size after augmentation, N_{y_i} is the number of samples of which label is equal to y_i .

H. Prediction

After finishing the k -th task, for each test sample x , we calculate the similarity between the embedding of x and each prototype embedding. For each label c we get:

$$p_c = \frac{1}{n_c} \sum_{i|y_i=c} \mathbf{E}(x_i)$$

If $g(\cdot)$ is Euclid, Manhattan distance:

$$y^* = \arg\min_{c=1..k} g(f(x), p_c)$$

Else if $g(\cdot)$ is cosine similarity:

$$y^* = \arg\max_{c=1..k} g(f(x), p_c)$$

IV. EXPERIMENTAL SETTINGS AND RESULTS

A. Datasets

Our experiments are run on two datasets:

- **FewRel** [13], [18]: It is a large-scale few-shot relation classification and relatively balanced dataset with 70,000 sentences on 100 relations [13]. Following up this work, version 2.0 of FewRel [18] proposed the problems of transferring models across different domains and detecting none-of-the-above relations.

Table I
COMPARE OUR MODEL WITH CRL AND RP-CRE MODELS ON FEWREL DATASET

FewRel										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
CRL	0.982	0.946	0.925	0.905	0.894	0.880	0.869	0.856	0.845	0.831
RP-CRE	0.978	0.951	0.918	0.905	0.899	0.877	0.866	0.856	0.844	0.826
CRL-protoAug	0.982	0.952	0.932	0.917	0.906	0.891	0.882	0.869	0.858	0.844

Table II
COMPARE OUR MODEL WITH CRL AND RP-CRE MODELS ON TACRED DATASET

TACRED										
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
CRL	0.978	0.932	0.898	0.847	0.841	0.813	0.802	0.791	0.790	0.780
RP-CRE	0.976	0.932	0.906	0.851	0.827	0.811	0.783	0.760	0.761	0.757
CRL-protoAug	0.978	0.939	0.918	0.872	0.861	0.848	0.829	0.820	0.819	0.810

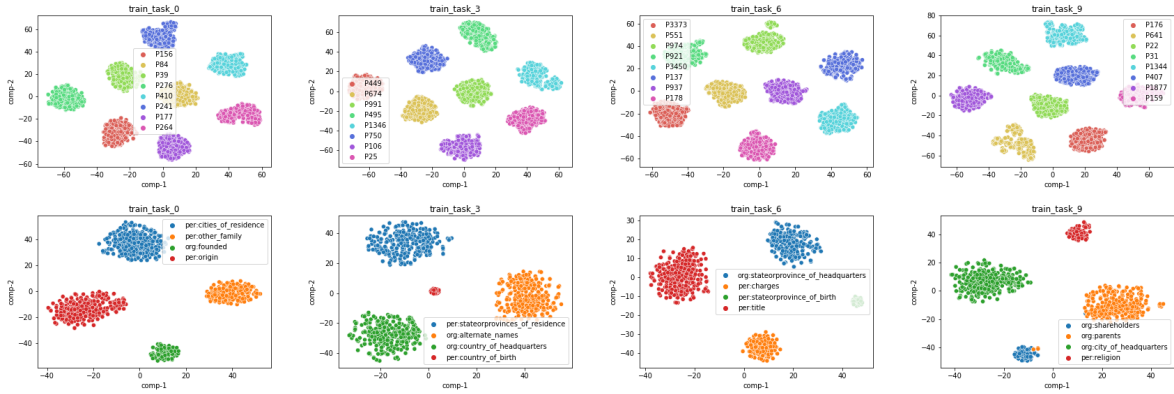


Fig. 2. A visualization of relation representation learned by our method at the different tasks on FewRel and TACRED dataset

- **TACRED** [19] is a large-scale supervised RE dataset with 106,264 examples, obtained via crowdsourcing and covering 41 TAC Knowledge Base Population relation types and *no_relation* label. TACRED is more imbalanced than FewRel.

B. Evaluation metrics and Baselines

For evaluating the efficiency of reducing catastrophic forgetting, we use average accuracy because of its emphasis on the performance of the model on earlier tasks [2], [17]. Moreover, following the suggestion from the above paper, we use the average accuracy of 10 tasks at each step.

We evaluate our combination of CRL and prototype augmentation and these two baselines, which have the best performance in CRE tasks at this time, with the above benchmarks for comparison:

- **RP-CRE** [17] achieves good performance in reducing catastrophic forgetting by refining sample embeddings. In this research, they save the prototype of each relation to extract relevant information from each relation in order to effectively leverage memorized samples. In particular, the prototype embedding for a given relation is calculated using examples of the relation, which

have been stored and gathered using the K-means algorithm.

- **CRL** [4] is proposed to maintain the relation embedding using contrastive learning and knowledge distillation. It is shown to perform better than other baselines at that time. This research trains each new task using supervised contrastive learning based on a memory bank so that the model can efficiently learn the relation representations. The samples are then contrastively replayed in memory. The model is made to retain its understanding of past relationships by memory knowledge distillation to mitigate forgetting of the previous task.

C. Result and Analysis

Table I and table II shows the result of the experiments. In this statistics table, T1, T2,..., T10 represent ten independent tasks, with Ti as the i-th task; the division is based on the idea of Zhao et al. in the original paper about CRL [4]. It is shown that our method outperforms the two baseline methods: CRL and RP-CRE, on both two datasets (FewRel and TACRED). This means that, without saving real data from old tasks, we augmentation data by a prototype of each relation to retrain in new tasks, and

the performance of the model even achieves higher performance than reusing real data.

With the FewRel dataset, in the first task, the CRL model and ours have a similar result with 98.2%, but then, the performance on the next task of CRL decrease significantly to 94.6%, and CRL just has 83.1% accuracy while our model has 84.4% on task 10, higher 1.3% than CRL model. In addition, the RP-CRE model gets lower results on ten tasks. Likewise, with TACRED dataset, both CRL and our models begin with 97.8 %. Afterward, the performance of CRL on task 2 declined remarkably to 93.2% and results on the last task are only 78% while our model still reached 81% on task 10. Compared with the RP-CRE model, all tasks get higher results from our model. The results show that our method produces the best representation for continual relaxation extraction.

Figure 2 illustrates that after the training process, the representations of data points are still good; specifically, the data points are close to each other in the same relation. Combined with the experimental results in Table I and Table II, our model performs stability on relations, which means this model alleviates well the catastrophic forgetting problem.

V. CONCLUSIONS AND FUTURE WORK

This paper proposes a novel consistent representation learning method for CRE tasks, mainly by combining the advantages of stability of CRL and generating pseudo-samples by a prototype of each relation to adding in memory bank instead of using real samples. Experiments illustrate that our approach performs more stability, thus better than the two best baselines at this time in alleviating catastrophic forgetting. In the future, we will continue to research how to select representative samples after training to upgrade the result of reducing the catastrophic forgetting problem.

ACKNOWLEDGMENT

This work has been supported by VNU University of Engineering and Technology under project number CN22.07.

REFERENCES

- [1] Huang, Yi Yao, and William Yang Wang. "Deep residual learning for weakly-supervised relation extraction." *arXiv preprint arXiv:1707.08866* (2017).
- [2] Han, Xu, et al. "Continual relation learning via episodic memory activation and reconsolidation." *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.
- [3] Wang, Hong, et al. "Sentence embedding alignment for life-long relation extraction." *arXiv preprint arXiv:1903.02588* (2019).
- [4] Kang Zhao, Hua Xu, Jiangong Yang, and Kai Gao. 2022. Consistent Representation Learning for Continual Relation Extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3402–3411, Dublin, Ireland. Association for Computational Linguistics.
- [5] 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", in *proceedings of the National Academy of Sciences*, 2017, pages 3521–3526.
- [6] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Continual learning through synaptic intelligence", in *International Conference on Machine Learning*, 2017, pages 3987–3995.
- [7] Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. "Net2net: Accelerating learning via knowledge transfer", *arXiv preprint arXiv:1511.05641* (2015).
- [8] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. "Pathnet: Evolution channels gradient descent in super neural networks", *arXiv preprint arXiv:1701.08734* (2017).
- [9] David Lopez-Paz and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning", *advances in Neural Information Processing Systems*, 2017, pages 6467–6476.
- [10] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. "Efficient lifelong learning with a-gem", *arXiv preprint arXiv:1812.00420* (2018).
- [11] Fan-Keng Sun, Cheng-Hao Ho, Hung-Yi Lee. "LAMOL: LAnguage MOdeling for Lifelong Language Learning", in *Proceedings of International Conference on Learning Representations*, 2020, doi: 10.48550/arXiv.1909.03329.
- [12] Zhu, Fei, et al. "Prototype augmentation and self-supervision for incremental learning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.
- [13] Han, Xu, et al. "Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation." *arXiv preprint arXiv:1810.10147* (2018).
- [14] de Masson D'Autume, Cyprien, et al. "Episodic memory in lifelong language learning." *Advances in Neural Information Processing Systems* 32 (2019).
- [15] Jaiswal, Ajay, et al. "SCALP-Supervised Contrastive Learning for Cardiopulmonary Disease Classification and Localization in Chest X-rays using Patient Metadata." *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021.
- [16] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, Dilip Krishnan. "Supervised Contrastive Learning", presented at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada, Dec. 6-12, 2020.
- [17] Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. "Refining sample embeddings with relation prototypes to enhance continual relation extraction", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pages 232–243.
- [18] Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. "FewRel 2.0: Towards More Challenging Few-Shot Relation Classification", in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pages 6250–6255.
- [19] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. "Position-aware attention and supervised data improve slotfilling". In *proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pages 35–45.
- [20] Soares, Livio Baldini, et al. "Matching the blanks: Distributional similarity for relation learning." *arXiv preprint arXiv:1906.03158* (2019).



Original Article

Contrastive Learning for Boosting Knowledge Transfer in Task-Incremental Continual Learning of Aspect Sentiment Classification Tasks

Thanh Hai Dang*, Quynh-Trang Pham Thi, Duc-Trong Le, Tri-Thanh Nguyen

VNU University of Engineering and Technology, 144 Xuan Thuy, Cau Giay, Hanoi, Vietnam

Received 29 November 2023;

Revised 04 May 2024; Accepted 15 May 2024

Abstract: Continual learning (CL) aims to learn a sequence of tasks, with task datasets emerging incrementally over time, and without a predetermined number of tasks. CL models strive to achieve two primary objectives: preventing catastrophic forgetting and facilitating knowledge transfer between tasks. Catastrophic forgetting refers to the sharp decline in the performance of CL models on previously learned tasks as new ones are learned. Knowledge transfer, which leverages acquired knowledge from previous tasks, empowers the CL model to adeptly tackle new tasks. However, only a few CL models proposed by far successfully achieve those two objectives simultaneously. In this paper, we present a task-incremental CL based model that leverages a pre-trained language model (i.e., BERT) with injected CL-plugins to mitigate catastrophic forgetting in continual learning. Additionally, we propose the utilization of two contrastive learning-based losses, namely contrastive ensemble distillation (CED) and contrastive supervised learning of the current task (CSC) losses, to enhance our model's performance. The CED loss improves the knowledge transferability of our continual learning model, while the CSC loss enhances its performance for the current learning task. Experimental results on benchmark datasets demonstrate that our proposed model outperforms all existing continual learning models in the task-incremental learning setting for continual aspect sentiment classification.

Keywords: Continual Learning, Contrastive Learning, Aspect-Sentiment Classification.

1. Introduction

Continual learning (CL) is designed to learn a sequence of tasks where task datasets become

available progressively over time, and the number of tasks is not predetermined [1]. This learning paradigm allows leveraging acquired knowledge from past tasks without storing previously trained

*Corresponding author

E-mail address: hai.dang@vnu.edu.vn

<https://doi.org/10.25073/2588-1086/vnucsce.1833>

data, thereby addressing security concerns. There are three popular settings for continual learning systems [2], namely Task incremental learning (TIL), Domain incremental learning (DIL) and Class incremental learning (CIL). TIL needs to know which task it is handling in the testing phase. TIL builds a prediction component specific for each task within one unified network. During the inference, it is routed to the task-specific component within the learned model based on the provided task identity. In contrast to TIL, DIL does not require knowing the task identity when testing. DIL aims to learn a single task, but in different contexts. Both TIL and DIL demand that when learning a new task the data labels must be seen in the previous tasks. CIL relaxes this strict requirement. It can automatically learn incoming data of a new class/label that did not appear when learning the previous tasks [3].

CL encounters a significant obstacle known as catastrophic forgetting, wherein the model's parameters being already learned for previous tasks must be readjusted to optimize performance for a new task. As a consequence, this significantly diminishes the model's effectiveness on previously learned tasks. Early CL models [4] based on fine-tuning a pre-trained language model BERT [5] subsequently for coming tasks are severe victims of this parameters' readjustment [1, 6–8]

To mitigate catastrophic forgetting, a natural approach is to minimize reconfiguration of models' parameters as much as possible. To this end, various CL models have been proposed, of which each task has its own specific module (or subset of the model's parameters) to be optimized. When learning a new task, only the module specific for this task is tuned while others remain untouched. These modules can be task-specific adapters [9, 10] or continual learning plugins (CL-plugin) [11] or a relatively small 2-layer fully connected network (so-called a Capsule Networks) [12, 13], which are injected

into the pre-trained BERT.

The isolation of task-specific modules helps CL models mitigate catastrophic forgetting. However, it hinders (or even prohibit from) the shared knowledge utilization (update included) among tasks, which can be useful for enhancing the model's performance on related tasks. Knowledge utilization enables CL models to leverage insights acquired from past learned tasks in one domain and apply them effectively to new tasks in other domains. Without effective knowledge exploitation mechanisms, CL models may struggle to adapt to the nuances of different domains, resulting in reduced performance and reliability. This contradiction causes CL hard to achieve both objectives together, namely preventing catastrophic forgetting and facilitating knowledge transfer. They are still two significant challenges for continual learning.

To our knowledge, only a few proposed CL models successfully achieve those two objectives at the same time. Among them, CTR [11] and CLASSIC [14] recently proposed are two well-established CL models. The former belongs to the TIL setting while the latter is of the DIL setting. Both are dedicated to the Aspect Sentiment Classification (ASC) tasks. ASC is a task that identifies a sentiment about the aspect of an object whether positive, negative, or neutral. For example, the sentence "the mic quality is quite nice" is an opinion about the aspect of "mic quality" of a mobile phone object. This sentence should be classified as a "Positive" sentiment by an ASC model.

Given a text, Aspect Sentiment Classification (ASC) involves determining whether the sentiment towards a specific aspect of an object is positive, negative, or neutral. For instance, the sentence "the mic quality is quite nice," expresses the sentiment pertaining to the aspect "mic quality" of a mobile phone. An ASC model should correctly classify this sentence as conveying a "Positive" sentiment. In the realm of Aspect Sentiment Classification (ASC) continual

learning, discerning sentiment nuances are about various aspects and/or objects of totally different evolving contexts and domains.

In the realm of Aspect Sentiment Classification (ASC) continual learning, the challenges of catastrophic forgetting and insufficient knowledge transfer are particularly pronounced due to the intricate nature of sentiment analysis tasks. ASC models, tasked with discerning sentiment nuances across various aspects and domains, confront a relentless stream of new data and evolving contexts. Inevitably, this causes ASC continual learning models to suffer from severe catastrophic forgetting.

Even when domains for ASC tasks largely differ, sentiment expressions inherent to one domain still share some commonalities with those in other domains, as they are all conveyed using the same language. Therefore, the importance of knowledge transfer in Continual Learning for ASC becomes even more pronounced and crucial for ensuring robust adaptation and accurate sentiment analysis, particularly when dealing with disparate domains. This process allows a CL model to grasp domain-specific sentiment patterns and nuances more efficiently, leading to improved accuracy and generalization across diverse domains.

In this work, we explore how much knowledge should be transferred in continual learning and empirically demonstrate its effectiveness for sequential aspect sentiment classification (ASC) tasks in the TIL setting. To this end, we propose a continual ASC model in the TIL setting that selectively transfers knowledge across ASC tasks. This is achieved by utilizing a normalized cross-attention contrastive loss in the classification head of the model. In addition, our model enhances its performance on the current task through contrastive supervised learning (CSC).

To evaluate our model, we conduct extensive experiments on 19 benchmark ASC data sets from various domains. Experimental results

indicate that our model outperforms recent state-of-the-art continual learning ASC models in the TIL setting, thus demonstrating the effectiveness of our model in transferring selective distilled knowledge between tasks for continual learning.

The rest of this paper is organized as follows: Section 2 reviews the related work; Section 3 presents our proposed method; Section 4 details the experimental results; and the final section concludes with future directions.

2. Related Work

Various advanced approaches have been proposed for continual learning to mitigate CF [1], such as Regularization-based approach [15] and Memory (or Replay) based approach [16–18]. The former estimates the importance of parameters for previously learned tasks to penalize changes in these parameters to avoid catastrophic forgetting. The latter stores a few samples of all previous tasks and then exert these in-memory data to update the parameters of the current task. As a result, it helps CL models retain the knowledge of the previous tasks, thereby mitigating catastrophic forgetting.

Recent years have seen a surge of research efforts focusing on the integration of advanced pre-trained models, such as BERT (Bidirectional Encoder Representations from Transformers) [5], into continual learning approaches. BERT adapters, lightweight task-specific neural network components, have emerged as a promising solution [9], which chooses to freeze the parameters of the pre-trained BERT, only tuning a small number of parameters of injected task-specific adapters [9–11]. A BERT adapter can be a relatively small 2-layer fully connected network or a Capsule Networks (CapsNet) [12, 13] that uses vector capsules instead of scalar feature detectors. These task-specific adapters enable CL models to learn new tasks while preserving knowledge from previous tasks. Researchers have explored combining regularization and

memory-based techniques with BERT adapters to further enhance their performance in continual learning scenarios. Regularization methods are applied to adapter parameters to mitigate forgetting, while memory-based strategies such as Experience Replay and Generative Replay are integrated to retain knowledge from past tasks. By leveraging the strengths of both pre-trained models and continual learning techniques, these approaches aim to develop robust and adaptable models across diverse tasks and domains.

Although the CL capability of existing advanced models have been empirically demonstrated they still struggle with CF issues when tasks lack substantial shared knowledge [11, 14].

Isolating task-specific modules in CL models aids in mitigating catastrophic forgetting to some extent. However, it poses a challenge by limiting the utilization and updating of shared knowledge across tasks, which could enhance the model's performance on related tasks. This contradiction makes it difficult for CL to simultaneously achieve both objectives: preventing catastrophic forgetting and facilitating knowledge transfer.

As far as we know, only a handful of continual learning (CL) models have effectively achieved both objectives simultaneously. Notably, CTR [11] and CLASSIC [14] are two prominent CL models that have recently demonstrated success in this regard. Both models employ task masks to isolate task-specific knowledge for dealing with CF. While CTR operates within the TIL setting, CLASSIC is designed for the DIL scenario. Remarkably, both models are specifically tailored for Aspect Sentiment Classification (ASC) tasks.

CLASSIC introduces an approach for knowledge transfer through contrastive learning, focusing on domain continual learning (i.e. DIL setting). Its key innovation lies in a contrastive continual learning method facilitating both knowledge transfer across tasks and distillation from old tasks to the new task. CLASSIC

utilizes BERT-Adapter [9], maintaining BERT parameters unchanged while achieving performance comparable to BERT fine-tuning. By proposing task masks to isolate task-specific knowledge and a contrastive continual learning method for knowledge transfer and distillation, CLASSIC significantly enhances the accuracy of all tasks.

CTR utilizes a key component known as the CL-plugin that is injected into BERT at two locations. Comparing with BERT-Adapter, CL-plugins differ significantly. While an adapter is a simple 2-layer fully-connected network inserted into BERT (to be fine-tuned) for each specific end task, the CL-plugin functions as a capsule network, which deviates from traditional neural networks by utilizing vector-output capsules instead of scalar activations, thereby preserving additional information in a more nuanced manner. CL-plugin integrates a novel transfer routing mechanism that facilitates knowledge exchange across tasks while safeguarding task-specific information to prevent interference. CTR can learn all tasks using only one pair of CL-plugin modules inserted into BERT. By this strategical integration, CTR eliminates the need for individual fine-tuning of BERT for each task, which often leads to catastrophic forgetting [11].

3. Method

Taking inspiration from [11], we employ a continual learning plugin (CL-plugin) into our model within the TIL setting. The CL-Plugin comprises two essential modules, namely Knowledge Sharing Sub-Module (KSM) and Task-Specific Sub-Module (TSM). KSM identifies and transfers shareable knowledge from analogous previous tasks to the new task and TSM focuses on learning task-specific neurons and their associated masks. These masks serve to protect the neurons from updates by future tasks, thereby addressing the challenge of catastrophic forgetting (CF). Further, we

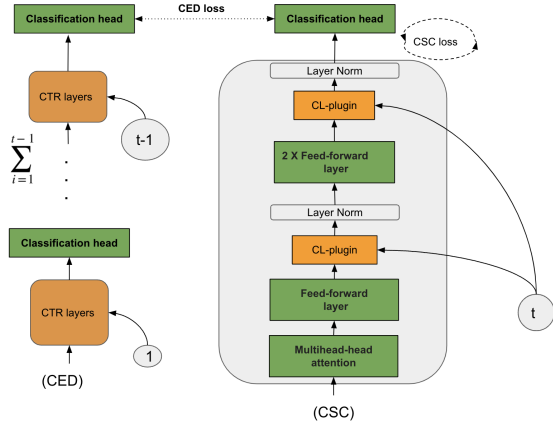


Figure 1. Overall architecture of our proposed model.

propose to boost the TIL performance of our CL-plugin based model by integrating contrastive learning losses [14] into classification heads of our model, including contrastive ensemble distillation (CED) loss to capture shared knowledge across tasks and contrastive supervised learning on the current task (CSC) loss to further enhance task-specific performance. The CED loss encourages knowledge transfer further by distilling knowledge from all previous tasks into the current task. The CSC loss aims to improve the plasticity of the current task.

Fig. 1 illustrates the overall architecture of our proposed method. The detailed components are presented as follows.

3.1. Continual Learning Plugin (CL-plugin)

Given hidden states $h(t)$ extracted from the feed-forward layer within a transformer layer, and the task ID t , the CL-plugin yields outputs that comprise hidden states with informative features specific to the t^{th} task. Finally, CL-plugin employs the cross-entropy loss to optimize the model's parameters, expressed as follows:

$$\mathcal{L}_{CE} = \sum_{i=1}^N -y_i \log \hat{y}_i \quad (1)$$

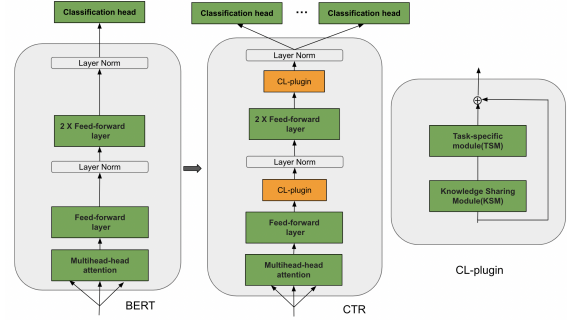


Figure 2. General architecture of BERT (left) and the CTR system (middle) and the CL-plugin (right) that are integrated into BERT [11].

where $\hat{y}_i$ is the probability score of the i^{th} sample in the batch data after passing through the model.

Inside the CL-plugin, the KSM consists of two capsule layers, namely the task capsule layer and the knowledge-sharing capsule layer, equipped with a dynamic routing algorithm. This configuration effectively clusters similar tasks and shared knowledge, enabling knowledge transfer among tasks with commonalities. In contrast, the TSM comprises differentiable fully-connected layers, with each layer's output undergoing additional processing using a task-specific mask. This mask indicates which neurons need protection for the specific task to address CF, preventing gradient updates on these neurons masked for the specific task during back-propagation when adapting to a new task (see Fig. 2). Specially, for a specific task ID t , $e_l^{(t)}$ denotes its embedding trained for the l^{th} layer within the TSM. This embedding encompasses differentiable parameters that can be learned concurrently with other components of the network. The task mask $m_l^{(t)}$, akin to a "soft" binary mask, is generated by employing a pseudo-gate function denoted as σ and incorporating a positive scaling hyper-parameter denoted as s throughout the training procedure. The calculation of $m_l^{(t)}$ is outlined as follows:

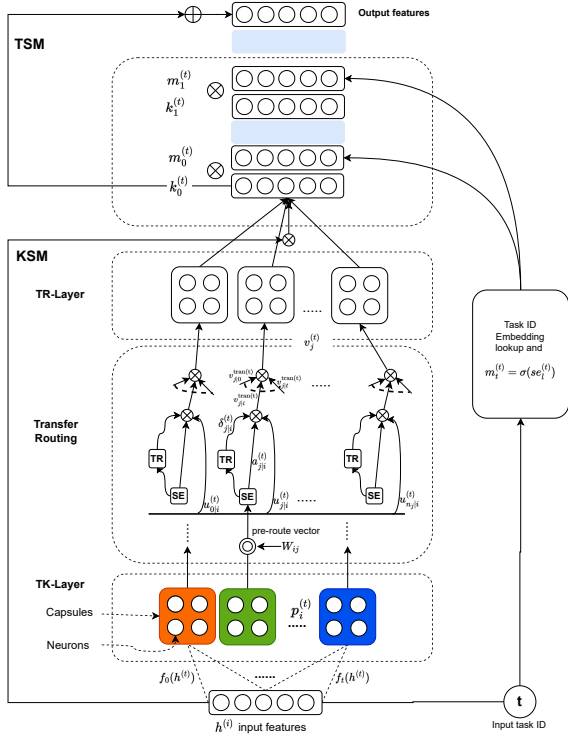


Figure 3. The architecture details of CL-plugin within our model.

$$m_l^{(t)} = \sigma(se_l^{(t)}) \quad (2)$$

For tasks sharing some common knowledge, their masks may have some neurons that coincide with each other. Each layer output $k_l^{(t)}$ in TSM when learning the task t is element-wise multiplied with $m_l^{(t)}$. To this end, the masked output $k^{(t)}$ from the last layer is then passed to the subsequent BERT model layer through a skip connection (see TSM in Fig. 3). The final $m_l^{(t)}$ for all layers after learning the task t are saved for further used.

3.2. Contrastive Ensemble Distillation (CED) Loss

This loss employs contrastive learning to distill knowledge from all previous tasks into the current task t . For each previous task i ,

the Contrastive Ensemble Distillation (CED) is calculated as follows:

$$\mathcal{L}_{CED}^{(i)} = \sum_{n=1}^N -\log \frac{\exp((z_n^{(i)} \cdot z_n^{(t)})/\tau)}{\sum_{j=1, j \neq n}^N \exp((z_n^{(i)} \cdot z_j^{(t)})/\tau)} \quad (3)$$

where N is the batch size; τ is an adjustable temperature parameter controlling the separation of classes; n is the index of the data sample in the batch; $z_n^{(i)}$ and $z_n^{(t)}$ are the logits for the same input data at the index n , generated from our model for the previous task i and current task t , respectively. This logit pair is treated as the positive pair and all of the other pairs are considered as negative pairs.

Since the model for each previous task remains fixed during the current task's training process, it effectively serves as a teacher for the current task (acting as the student). For $t-1$ models corresponding to all previous tasks, the final CED is formulated as Formula 4.

$$\mathcal{L}_{CED} = \sum_{i=1}^{t-1} \mathcal{L}_{CED}^{(i)} \quad (4)$$

3.3. Contrastive Supervised Learning of Current Task (CSC) Loss

Taking inspiration from CLASSIC [14], we formulate the Contrastive Supervised Learning of Current Task (CSC) Loss to enhance the performance of the current task, as follows:

$$\mathcal{L}_{CSC} = \sum_{n=1}^N \frac{1}{N_{y_n} - 1} \sum_{j=n, y_j=y_n}^N \log \frac{\exp\left(\frac{h_n^{(t)} \cdot h_j^{(t)}}{\tau}\right)}{\sum_{k=1, y_k \neq y_n}^N \exp\left(\frac{h_n^{(t)} \cdot h_k^{(t)}}{\tau}\right)} \quad (5)$$

Where N_{y_n} is the number of samples in the data batch that have the same label as y_n , while $h_j^{(t)}$ is the masked output of the TSM for the j^{th} sample in the batch, derived from the layer before the classification head of the task t model.

3.4. Model's Loss

We combine three losses described above to optimize our CL model in the TIL setting, as follows:

$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{CED} + \mathcal{L}_{CSC} \quad (6)$$

where $\mathcal{L}_{CE}$ denotes the cross-entropy loss, $\mathcal{L}_{CED}$ represents the contrastive ensemble distillation loss calculated using Formula 4 and $\mathcal{L}_{CSC}$ indicates the contrastive supervise learning of current task as expressed in Formula 5.

4. Experimental Results

We conducted experimental evaluations, comparing existing non-continual learning and task-incremental continual learning models for a sequence of ASC tasks on benchmark datasets. The reported results are averaged across five random task orders.

4.1. Experiment Datasets

Like other state-of-the-art continual learning models in the TIL setting for ASC tasks (such as CTR [11]), we employ four benchmark datasets, namely: (1) HL5Domains [19], comprising review sentences for 5 products; (2) Liu3Domains [20], focusing on 3 products; (3) Ding9Domains [21], covering 9 products; and (4) SemEval14 [22], relating to 2 products. These datasets collectively encompass sentiments about 19 aspects, each treated as an individual aspect-based sentiment classification task. Detailed statistics for each dataset are provided in Table 1.

In alignment with previous studies, we partitioned 10% of the original data for validation and another 10% for testing, for each dataset (1), (2), and (3). However, for dataset (4), we opted for a validation set consisting of only 150 examples from the training set. To maintain methodological consistency with prior research, we present a comprehensive breakdown of the sample numbers for each task in Table 1.

4.2. Hyperparameters

Our approach utilises a two-layer fully connected network with 768 dimensions in the capsule layers and three transfer capsules. Concerning the task-specific modules, we employ 2000 dimensions for the final states. To optimize our model, we use the Adam optimizer and set the learning rate to $3e-5$. We conduct training for ten epochs on the SemEval datasets and extend it to 30 epochs for other datasets. For all the baseline models, we utilize the code provided by their respective authors and adapt it for classification purposes.

4.3. Results and Analysis

Firstly, we employ multi-task learning, an approach that incorporates comprehensive knowledge of all preceding tasks and is trained on the entire dataset, to evaluate the upper bound of this problem. Then, we employ non-continual learning baselines independently for each task. We have three baselines of such, namely BERT, BERT (frozen), and Adapter-BERT. Continuously, we set the foundation based on "no forgetting handling" (NFH). To this end, we compare our model against 12 state-of-the-art continual learning models in the TIL setting, including KAN [23], SRK [24], HAT [25], CAT [6], UCL [26], EWC, L2 [15], OWM [27], A-GEM [28], DER++ [29], BCL [10], LAMOL [30].

Due to imbalanced classes, we calculate both Accuracy and Macro-F1 (MF1) scores to account for potential biases in performance evaluation. As shown in Table 2, our model demonstrates strong continual learning ASC performance, achieving an Accuracy of 88.5% and a Macro-F1 score of 82.35%. Notably, our model outperforms all baseline models, including state-of-the-art (SOTA) continual learning models, in terms of Macro-F1. While our model's Accuracy ranks second, trailing behind LAMOL by only 0.41%, it's worth noting that LAMOL is based on GPT-2, whereas our model is based on

Table 1. Number of sentences per dataset, referred to as tasks in the context of continual learning [11].

Data source	Task/domain	Train	Validation	Test
Liu3domain	Speaker	352	44	44
	Router	245	31	31
	Computer	283	35	36
HL5domain	Nokia6610	271	34	34
	Nikon4300	162	20	21
	Creative	677	85	85
	CanonG3	228	29	29
	ApexAD	343	43	43
Ding9domain	CanonD500	118	15	15
	Canon100	175	22	22
	Diaper	191	24	24
	Hitachi	212	26	27
	Ipod	153	19	20
	Linksys	176	22	23
	MicroMP3	484	61	61
	Nokia6600	362	45	46
	Norton	194	24	25
SemEval14	Rest.	3452	150	1120
	Laptop	2163	150	638

BERT. To ensure a fair comparison, we replicate experiments with CTR using the same task orders as those employed in our model experiments (Results denoted by the label CTR*). However, CTR* yields inferior results compared to our model in terms of both Accuracy and Macro-F1 metrics. It's worth noting that the Accuracy and Macro-F1 values reported in their paper are obtained from different task orders, which are not disclosed. Additionally, compared to the non-continual learning method Adapter-BERT, our continual learning model demonstrates a higher performance of 3% in both Accuracy and Macro-F1 metrics. In comparison to the non-continual learning approach of Adapter-BERT, our continual learning model achieves a superior performance, with a 3% increase in both Accuracy and Macro-F1 metrics.

4.4. Ablation Study

We conducted experiments by removing specific components from our method, and the results in Table 3 highlight the effectiveness

of the CL-plugin, CED, and CSC. The outcomes, averaged across two evaluation metrics (Accuracy and MF1) from 5 random task orders, demonstrate notable improvements. Specifically, when the CL-plugin is omitted, a two-layer fully connected adapter is employed. The results reveal that the inclusion of the CL-plugin enhances the model's performance by approximately 2.6% in both accuracy and MF1 score. Additionally, Table 3 illustrates the positive impact of incorporating two contrastive learning-based losses (CED and CSC) on the continual ASC performance of our model. We also conducted experiments with our model, incorporating another contrastive learning-based loss known as the Knowledge Sharing Loss (CKS), inspired by [14]. However, experimental results revealed that CKS did not yield any improvement for our model (data not shown). This could be attributed to the fact that our model already includes the KSM module, which facilitates knowledge sharing in a manner similar to CKS.

Table 2. Average Accuracy (Acc.) and Macro-F1 (MF1) over five random sequences of 19 tasks.

Scenario	Category	Model	Acc.(%)	MF1(%).
Non-continual Learning (SDL)	BERT	MTL	91.91	88.11
	BERT	SDL	85.84	76.35
	BERT (Frozen)	SDL	78.14	58.13
	Adapter-Bert	SDL	85.96	78.07
	W2V	SDL	77.01	51.89
Continual Learning	BERT	NFH	49.60	43.08
	BERT (Frozen)	NFH	85.51	76.64
	Adapter-BERT	NFH	85.51	76.64
	W2V	NFH	82.69	73.56
	BERT (frozen)	L2	56.04	38.40
		A-GEM	86.06	78.44
		DER++	84.27	75.08
		KAN	85.49	77.38
		SRK	84.76	78.52
		EWC	86.37	74.52
		UCL	83.89	74.82
		OWM	87.02	79.31
		HAT	86.74	78.16
		CAT	83.68	68.64
	Adapter-BERT	L2	63.97	52.43
		A-GEM	45.88	28.21
		DER++	47.63	35.54
		EWC	56.30	49.58
		UCL	64.46	36.64
		OWM	72.99	66.51
		HAT	86.14	78.52
		BCL	88.29	81.40
		LAMOL	88.91	80.59
		CTR	89.47	83.62
		CTR*	88.21	81.19
		Ours	88.50	82.35

Table 3. Ablation experimental results of our model.

Model	Acc.(%)	MF1(%)
Ours	88.50	82.35
without CL-plugin	85.83	79.75
without CSC+CED	86.48	79.32
without CSC	87.52	79.93
without CED	87.98	80.10

5. Conclusion

This paper investigates the task-incremental learning (TIL) paradigm for the continual learning of a sequence of ASC tasks. Our approach leverages a CL-plugin to capitalize on the acquired knowledge from a pre-trained language model, i.e. BERT, mitigating catastrophic forgetting in continual learning. Additionally, we propose the utilization of two contrastive learning-based losses, namely contrastive ensemble distillation (CED) and contrastive supervised learning of the current task (CSC) losses, to enhance our model's performance. The CED loss improves the knowledge transferability of our continual learning model, while the CSC loss enhances its performance for the current learning task. Our proposed model outperforms all existing continual learning models in the task-incremental learning setting for continual aspect sentiment classification. For future research directions, exploring the application of memory-based methods to address catastrophic forgetting could further enhance the effectiveness of our model.

References

- [1] Z. Chen, B. Liu, Lifelong machine learning, Synthesis Lectures on Artificial Intelligence and Machine Learning, Vol. 12, No. 3, 2018, pp. 1–207.
- [2] G. M. van de Ven, A. S. Tolias, Three scenarios for continual learning, CoRR, Vol. abs/1904.07734, (2019). arXiv:1904.07734.
URL <http://arxiv.org/abs/1904.07734>
- [3] S. Rebuffi, A. Kolesnikov, C. H. Lampert, icarl: Incremental classifier and representation learning, CoRR, Vol. abs/1611.07725, (2016). arXiv:1611.07725.
URL <http://arxiv.org/abs/1611.07725>
- [4] H. Xu, B. Liu, L. Shu, P. S. Yu, BERT post-training for review reading comprehension and aspect-based sentiment analysis, CoRR, Vol. abs/1904.02232, (2019). arXiv:1904.02232.
URL <http://arxiv.org/abs/1904.02232>
- [5] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR, Vol. abs/1810.04805, (2018). arXiv:1810.04805.
URL <http://arxiv.org/abs/1810.04805>
- [6] Z. Ke, B. Liu, X. Huang, Continual learning of a mixed sequence of similar and dissimilar tasks, CoRR, Vol. abs/2112.10017, (2021). arXiv:2112.10017.
URL <https://arxiv.org/abs/2112.10017>
- [7] Y. Huang, Y. Zhang, J. Chen, X. Wang, D. Yang, Continual learning for text classification with information disentanglement based regularization, CoRR, Vol. abs/2104.05489, (2021). arXiv:2104.05489.
URL <https://arxiv.org/abs/2104.05489>
- [8] M. Biesialska, K. Biesialska, M. R. Costa-jussà, Continual lifelong learning in natural language processing: A survey, CoRR, Vol. abs/2012.09823, (2020). arXiv:2012.09823.
URL <https://arxiv.org/abs/2012.09823>
- [9] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for NLP, CoRR, Vol. abs/1902.00751, (2019). arXiv:1902.00751.
URL <http://arxiv.org/abs/1902.00751>
- [10] Z. Ke, H. Xu, B. Liu, Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks, CoRR, Vol. abs/2112.03271, (2021). arXiv:2112.03271.
URL <https://arxiv.org/abs/2112.03271>
- [11] Z. Ke, B. Liu, N. Ma, H. Xu, L. Shu, Achieving forgetting prevention and knowledge transfer in continual learning, CoRR, Vol. abs/2112.02706, (2021). arXiv:2112.02706.
URL <https://arxiv.org/abs/2112.02706>
- [12] S. Sabour, N. Frosst, G. E. Hinton, Dynamic routing between capsules, CoRR, Vol. abs/1710.09829, (2017). arXiv:1710.09829.
URL <http://arxiv.org/abs/1710.09829>
- [13] G. E. Hinton, A. Krizhevsky, S. D. Wang, Transforming auto-encoders, in: International conference on artificial neural networks, Springer, 2011, pp. 44–51.
- [14] Z. Ke, B. Liu, H. Xu, L. Shu, CLASSIC: Continual and contrastive learning of aspect sentiment classification tasks, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 6871–6883. doi:10.18653/v1/2021.emnlp-main.550.
- [15] J. Kirkpatrick, R. Pascanu, N. C. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, CoRR, Vol. abs/1612.00796, (2016). arXiv:1612.00796.
URL <http://arxiv.org/abs/1612.00796>

- [16] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, CoRR, Vol. abs/1606.04671, (2016). arXiv:1606.04671.
URL <http://arxiv.org/abs/1606.04671>
- [17] J. Yang, K. Zhou, Y. Li, Z. Liu, Generalized out-of-distribution detection: A survey, CoRR, Vol. abs/2110.11334, (2021). arXiv:2110.11334.
URL <https://arxiv.org/abs/2110.11334>
- [18] S. Hou, X. Pan, C. C. Loy, Z. Wang, D. Lin, Learning a unified classifier incrementally via rebalancing, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [19] M. Hu, B. Liu, Mining and summarizing customer reviews, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, Association for Computing Machinery, New York, NY, USA, 2004, p. 168–177. doi:10.1145/1014052.1014073.
URL <https://doi.org/10.1145/1014052.1014073>
- [20] T. Liu, L. Ungar, J. Sedoc, Continual learning for sentence representations using conceptors, ArXiv, Vol. abs/1904.09187, (2019).
- [21] X. Ding, B. Liu, P. S. Yu, A holistic lexicon-based approach to opinion mining, in: Web Search and Data Mining, 2008.
- [22] Q. Jiang, L. Chen, R. Xu, X. Ao, M. Yang, A challenge dataset and effective models for aspect-based sentiment analysis, in: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), 2019, pp. 6280–6285.
- [23] Z. Ke, B. Liu, H. Wang, L. Shu, Continual learning with knowledge transfer for sentiment classification, CoRR, Vol. abs/2112.10021, (2021). arXiv:2112.10021.
URL <https://arxiv.org/abs/2112.10021>
- [24] G. Lv, S. Wang, B. Liu, E. Chen, K. Zhang, Sentiment Classification by Leveraging the Shared Knowledge from a Sequence of Domains, 2019, pp. 795–811. doi:10.1007/978-3-030-18576-3-47.
- [25] J. Serrà, D. Surís, M. Miron, A. Karatzoglou, Overcoming catastrophic forgetting with hard attention to the task, CoRR, Vol. abs/1801.01423, (2018). arXiv:1801.01423.
URL <http://arxiv.org/abs/1801.01423>
- [26] H. Ahn, D. Lee, S. Cha, T. Moon, Uncertainty-based continual learning with adaptive regularization, CoRR, Vol. abs/1905.11614, (2019). arXiv:1905.11614.
URL <http://arxiv.org/abs/1905.11614>
- [27] W. Xue, T. Li, Aspect based sentiment analysis with gated convolutional networks, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 2514–2523. doi:10.18653/v1/P18-1234.
URL <https://aclanthology.org/P18-1234>
- [28] A. Chaudhry, M. Ranzato, M. Rohrbach, M. Elhoseiny, Efficient lifelong learning with A-GEM, CoRR, Vol. abs/1812.00420, (2018). arXiv:1812.00420.
URL <http://arxiv.org/abs/1812.00420>
- [29] P. Buzzega, M. Boschini, A. Porrello, D. Abati, S. Calderara, Dark experience for general continual learning: a strong, simple baseline, ArXiv, Vol. abs/2004.07211, (2020).
- [30] F.-K. Sun, C.-H. Ho, H. yi Lee, Lamol: Language modeling for lifelong language learning, in: International Conference on Learning Representations, 2019.



ZeroRFF: A Random Fourier Features and Analytic Learning Method for Generalized Class Incremental Learning

Duc-Hung Nguyen, Tri-Thanh Nguyen, Thanh-Hai Dang,
and Quynh-Trang Pham Thi^(*)

Vietnam National University, University of Engineering and Technology,
Hanoi, Vietnam
{22021109, ntthanh, hai.dang, trangptq}@vnu.edu.vn

Abstract. Continual learning aims to train models across multiple tasks, each containing different knowledge. The biggest challenge in continual learning is that models quickly lose previously learned knowledge when acquiring new tasks, as they lack access to their old knowledge. Generalized Class Incremental Learning (GCIL) poses the challenge that knowledge may not be uniform and may overlap between tasks. Some current methods store some old knowledge to prevent the model from forgetting, but this can violate data security. This research proposes the ZeroRFF (Zero-Memory Random Fourier Features) method, which solves the generalized class incremental learning problem without storing old knowledge (exemplar-free). This method uses analytic learning techniques; instead of relying on gradient descent, ZeroRFF finds optimal solutions by directly solving equations where the derivative equals zero. Additionally, ZeroRFF integrates Random Fourier Features (RFF) to transform linearly non-separable data into a new space where the data becomes linearly separable, enhancing the model's classification capability. Experimental results on the CIFAR100 dataset show that the ZeroRFF method achieves superior performance compared to other methods.

Keywords: continual learning · exemplar-free · analytic learning · Random Fourier Features

1 Introduction

Catastrophic forgetting is a major phenomenon in continual learning [1, 2]. This phenomenon occurs when trained models lose knowledge about previous tasks or classes when learning new information. For example, if an image recognition model is trained to recognize new animal species, it may forget how to identify previously learned species. The main cause of this phenomenon is the fundamental nature of gradient-based training algorithms, which tend to focus

on and exhibit bias toward recent tasks (*task-recency bias*). Specifically, model predictions often prioritize classes from new tasks, reducing performance on old classes [3].

To illustrate, consider a student learning foreign languages. If the student only focuses on learning French after having learned English, without reviewing English, their English proficiency may significantly decline. Similarly, in continual learning, models require special mechanisms—this is precisely the goal of the continual learning problem: ensuring that models can still acquire new knowledge while simultaneously "reviewing" to avoid forgetting old knowledge.

In traditional Class-Incremental Learning (CIL) scenarios, tasks are assumed to have a fixed number of samples, and classes in each task are completely separate from previous tasks. However, this assumption does not reflect reality, where training data often includes both new classes and previously encountered classes, while the number of samples in each task may vary unevenly. A new scenario studied recently, which is closer to this reality, is called Generalized Class-Incremental Learning (GCIL) [4,5].

The uneven distribution of sample numbers and classes between tasks in GCIL can exacerbate the problem of *catastrophic forgetting*. In particular, samples belonging to minority classes in a data batch may be overlooked, leading to incomplete model learning during training. For example, in an e-commerce product recognition system, new data may include both popular products (such as smartphones) and less popular products (like specialised accessories). If the model prioritises popular classes due to their larger sample sizes, it may overlook or poorly represent less popular classes, thereby reducing overall performance.

To mitigate the phenomenon of *catastrophic forgetting* in continual learning, a simple yet effective method is to use memory storage mechanisms (replay-based methods). Memory storage methods [6] store a small number of samples from old classes, allowing the model to review previous knowledge while learning new information, thereby minimizing knowledge forgetting. However, this mechanism poses risks related to data privacy due to the storage of old samples.

This research focuses on solving the Generalized Continual Learning (GCIL) problem, a recently studied continual learning scenario where training data includes both new classes and previously encountered classes, with uneven sample distribution across tasks. To overcome challenges related to *catastrophic forgetting* and data privacy limitations, the method proposed in this study employs Analytic Continual Learning (ACL), a branch of Exemplar-Free Class-Incremental Learning (EFCIL) that does not require storing old data samples. This not only helps minimize privacy risks but also meets computational resource efficiency requirements. Furthermore, Analytic Continual Learning differs from many popular gradient descent-based training methods in that it does not seek minima by using iterative operations to gradually approach the target point, but instead finds optimal solutions by solving the equation where the derivative equals zero. To enhance the model's learning capability, the method integrates Random Fourier Features (RFF), a technique that transforms original feature vectors from a linearly non-separable space to a higher-dimensional space. In

this new space, data becomes linearly separable, allowing the model to learn more effective representations and improve classification performance for complex classes. The effectiveness of the method is evaluated through several standard metrics in the continual learning field, including area under the accuracy curve, average accuracy, and final task accuracy. Experimental results demonstrate that the proposed method achieves superior performance compared to both replay-based methods and exemplar-free methods. This confirms the potential of combining ACL and RFF in building flexible continual learning systems suitable for real-world scenarios with complex and continuously changing data.

2 Preliminaries

2.1 Generalized Class Incremental Learning

Class-Incremental Learning (CIL) is a common scenario in continual learning, where the model learns new classes in stages, with the assumption that classes in each task are completely disjoint and the number of samples in each task is fixed, as illustrated in Fig. 1. However, this assumption often does not accurately reflect real-world scenarios, where task data may include both new classes and previously encountered classes, while the number of samples for each class is unevenly distributed across tasks.

Generalized Class-Incremental Learning (GCIL) is proposed as an extended scenario of CIL, addressing the limitations of CIL by allowing the model to handle data with characteristics more similar to reality. In GCIL, training data in a task may contain previously learned classes mixed with new classes, and the number of samples in each task is not fixed but changes flexibly. GCIL better reflects real-world scenarios, where data is not neatly organized into separate tasks but often appears in a mixed and unpredictable manner.

The main difference between GCIL and CIL lies in flexibility and the ability to handle complex data. While CIL requires clearly divided tasks, GCIL allows the model to learn under conditions closer to reality, where data continuously changes and does not follow fixed boundaries. However, the complexity of GCIL also increases the challenge of *catastrophic forgetting*, especially when minority classes may be overlooked during training due to uneven data distribution.

Specifically, the continual learning problem includes a sequence of continuous tasks $\mathcal{D} = \{\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_T^{\text{train}}\}$, where $\mathcal{D}_t^{\text{train}} = \{x_i, y_i\}_{i=1}^{N_t}$ is the training dataset of the t -th task, and T is the total number of tasks.

The training dataset $\mathcal{D}_t^{\text{train}}$ consists of N_t data pairs, y_i is the label of data x_i and y_i is a one-hot vector of size $|\mathcal{C}|$, where $\mathcal{C}$ is the label set. x_i is an input image with dimensions $c \times w \times h$. The goal of GCIL in the k -th task is to train a model using the dataset $\mathcal{D}_k^{\text{train}}$ and evaluate the performance of that model on the test dataset $\mathcal{D}_{1:k}^{\text{test}}$. Here, $\mathcal{D}_{1:k}$ denotes the merged dataset including tasks from 1 to k .

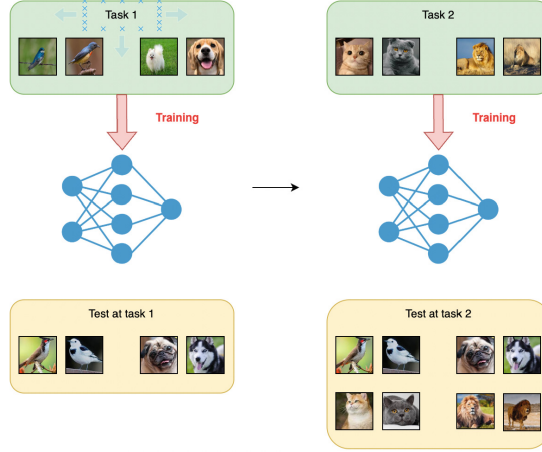


Fig. 1. Illustration of Class-Incremental Learning (CIL).

The Generalized Class-Incremental Learning (GCIL) scenario simulates real-world continual learning, where data distribution in terms of quantity and type can vary between tasks. In this scenario, the model is trained on a set of tasks, where each task may include classes belonging to the set $\mathcal{S}$, with a total number of classes being N . The number of samples of different classes in each task k is represented by a random vector $\mathbf{c}_k \in \mathbb{R}^N$. With $c_{k,i}$ being the i -th component of vector $\mathbf{c}_k$, representing the number of samples of class i in task k .

2.2 Analytic Continual Learning

Analytic Continual Learning (ACL) is an emerging branch in the field of Continual Learning, developed based on the foundation of Analytic Learning research [7]. Unlike traditional deep learning methods, which primarily rely on gradient descent algorithms to find optimal solutions for loss functions, ACL has a completely different approach. Specifically, instead of using iterative steps to adjust model parameters through gradients, ACL seeks to directly solve the equation where the derivative equals zero to determine the optimal solution. The ACIL method [8] transforms the continual learning problem into a least squares regression problem, eliminating the need for sample storage by maintaining correlation matrices. RanPAC [9] applies this technique to pre-trained models. GKEAL [10] focuses on continual learning scenarios with few samples (few-shot CIL) by using a Gaussian kernel projection. ACL is an emerging competitive branch in the field of continual learning.

2.3 Kernel Function

Kernel functions play an important role in solving nonlinear classification and regression problems. To handle data that is not linearly separable, we need a transformation to map the original data to a new space where the data becomes linearly separable. This transformation often creates data in a space with higher

dimensions, even infinite dimensions, compared to the original data. Direct computation of these functions can cause memory and computational performance issues. Instead of directly computing each data point in the new space, an efficient approach is to use kernel functions to describe the relationship between any two data points in the new space. Some popular kernel functions widely used in machine learning applications include:

- **Linear Kernel:** $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$. This is the basic case, suitable when data can be linearly separated.
- **Polynomial Kernel:** $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top \mathbf{y} + c)^d$, where $c \geq 0$ and d is the degree of the polynomial. This function allows modeling nonlinear relationships between features.
- **Radial Basis Function (RBF) Kernel:** $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$, where $\gamma > 0$. RBF is one of the most popular kernel functions, particularly effective in handling nonlinear data.
- **Sigmoid Kernel:** $K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x}^\top \mathbf{y} + c)$, with parameters κ and c chosen appropriately. This function originates from neural networks and can model nonlinear relationships.

3 Methodology

3.1 Overall Architecture

Figure 2 illustrates the architecture of our proposed method ZeroRFF, for Generalized Class-Incremental Learning (GCIL), which combines transformer-based feature extraction with analytic learning mechanisms to enable fast learning and uses a Fourier Features Module to enhance the model’s classification capability for sequential task learning.

The process begins with the input of task k , represented as $\mathcal{D}_k^{\text{train}} = \{X_k^{\text{train}}, Y_k^{\text{train}}\}$, containing training samples and their corresponding labels. The methodology is organized into four distinct phases, each serving a specific purpose in the continual learning pipeline.

Feature Extraction Phase: The input data first passes through a pre-trained DeiT-S/16 backbone network with fixed parameters Θ_{backbone} . This feature extractor, which has been pre-trained on ImageNet, transforms the raw input images into high-level feature representations $X_k^{(E)} = f_{\text{backbone}}(X_k^{\text{train}})$. The backbone remains frozen throughout the training process to prevent catastrophic forgetting at the feature level and ensure consistent representations across different tasks.

Kernel Mapping Phase: The extracted features $X_k^{(E)}$ are then transformed using Random Fourier Features (RFF) to obtain $X_k^{(B)} = \text{RFF}(X_k^{(E)})$. This transformation maps the input features into a higher-dimensional space where the data becomes approximately linearly separable. The RFF transformation is defined as $z(x) = \sqrt{\frac{2}{D}} \cos(\omega^T x + \beta)$, where $\omega \sim \mathcal{N}(0, \sigma^{-2} I)$ and $\beta \sim$

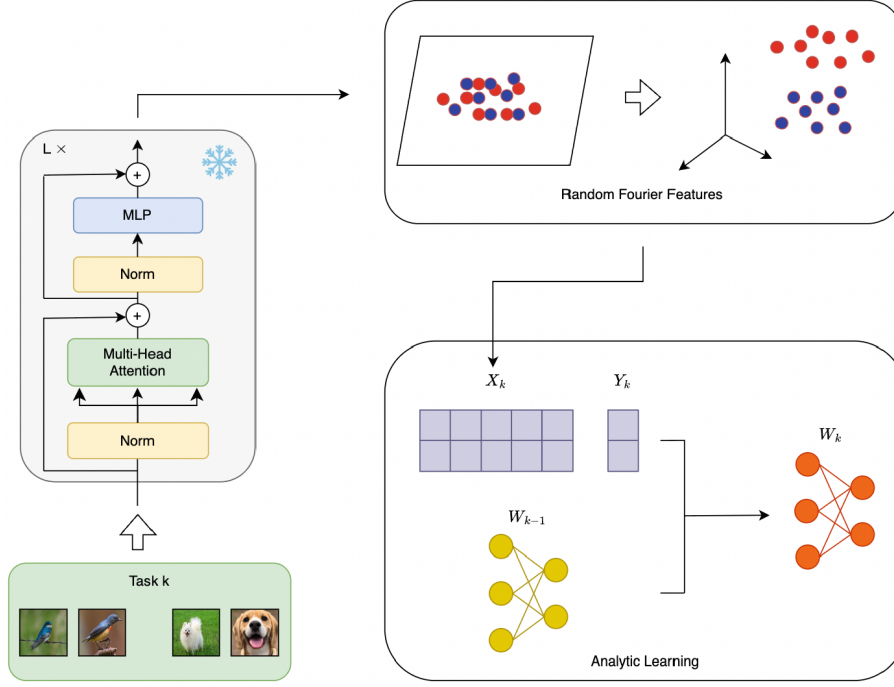


Fig. 2. Overall architecture of ZeroRFF.

$\text{Uniform}(0, 2\pi)$. With dimension $D = 5000$, this approximation effectively captures the behavior of the RBF kernel while avoiding the computational overhead of explicit kernel matrix calculations.

Exemplar-Free Learning Phase: The transformed features enter the analytic learning component, which represents the core innovation of our approach. Unlike traditional gradient-based optimization methods, this phase employs direct matrix solutions to find optimal parameters. The process maintains two key matrices: the correlation matrix R_k and the weight matrix $\hat{W}^{(k)}$. The correlation matrix is updated using the Sherman-Morrison formula for efficient recursive computation, while the weight matrix is updated through a direct analytical solution. Crucially, this phase requires no storage of previous task exemplars, addressing privacy concerns and reducing memory requirements.

Task Management Phase: After processing each task, the system checks whether all tasks have been completed. If additional tasks remain, the process increments the task counter ($k = k + 1$) and loads the next task dataset $\mathcal{D}_{k+1}^{\text{train}}$, creating a seamless loop for sequential task processing. Upon completion of all tasks, the methodology outputs the final model $\hat{W}_{\text{FCN}}^{(K)}$, which is ready for inference on the complete set of learned classes.

3.2 Algorithm Description

Algorithm 1 presents the complete ZeroRFF training procedure, which implements an exemplar-free approach to Generalized Class-Incremental Learning. The algorithm is designed to learn from a sequence of tasks without storing

previous task data, addressing both privacy concerns and memory constraints inherent in traditional continual learning approaches.

Input and Initialization: The algorithm takes as input a sequence of training tasks $\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_K^{\text{train}}$, where each task $\mathcal{D}_k^{\text{train}}$ consists of input samples X_k^{train} and their corresponding labels Y_k^{train} . Additionally, the algorithm requires a pre-trained backbone model with fixed weights Θ_{backbone} , typically a DeiT-S/16 transformer pre-trained on ImageNet. The initialization phase establishes two critical matrices: the correlation matrix $R_0 = \gamma I$, where γ is the regularization parameter, and the weight matrix $\hat{W}_{\text{FCN}}^{(0)} = 0$.

Sequential Task Processing: The core of the algorithm operates through a sequential loop over all tasks from $k = 1$ to K . For each task, the algorithm performs four essential operations that collectively enable exemplar-free continual learning.

The first operation (Line 3) involves feature extraction using the pre-trained backbone: $X_k^{(E)} \leftarrow f_{\text{backbone}}(X_k^{\text{train}}, \Theta_{\text{backbone}})$. This step transforms raw input images into high-level feature representations while maintaining consistency across tasks through the use of frozen pre-trained weights. The feature extraction process ensures that the input data is represented in a meaningful semantic space suitable for subsequent kernel transformations.

The second operation (Line 4) applies the Random Fourier Features transformation: $X_k^{(B)} \leftarrow \text{rff}(X_k^{(E)})$. This transformation maps the extracted features into a higher-dimensional space where the data becomes approximately linearly separable. The RFF transformation is crucial for enabling the analytic learning approach, as it provides a finite-dimensional approximation to the infinite-dimensional feature space induced by the RBF kernel.

Analytic Learning Updates: The third and fourth operations (Lines 5–6) implement the core analytic learning mechanism through recursive matrix updates. The correlation matrix update (Line 5) employs the Sherman-Morrison formula:

$$R_k \leftarrow R_{k-1} - R_{k-1} X_k^{(B)T} (I + X_k^{(B)} R_{k-1} X_k^{(B)T})^{-1} X_k^{(B)} R_{k-1} \quad (1)$$

This update maintains the inverse of the accumulated covariance matrix without explicitly storing or accessing previous task data. The Sherman-Morrison formula enables efficient rank-one updates to matrix inverses, making the computation tractable even for high-dimensional feature spaces.

The weight matrix update (Line 6) computes the optimal parameters using a direct analytical solution:

$$\hat{W}_{\text{FCN}}^{(k)} \leftarrow \hat{W}_{\text{FCN}}^{(k-1)} - R_{k-1} X_k^{(B)T} X_k^{(B)} \hat{W}_{\text{FCN}}^{(k-1)} + R_k X_k^{(B)T} Y_k^{\text{train}} \quad (2)$$

This recursive formulation ensures that the weight matrix incorporates information from all previous tasks while being updated solely based on the current task data. The direct analytical nature of this update eliminates the need for iterative optimization procedures, resulting in faster convergence and more stable learning dynamics.

Computational Efficiency: The algorithm’s computational complexity is dominated by matrix operations involving $D \times D$ matrices, where $D = 5000$ is the RFF dimension. The Sherman-Morrison update requires $O(D^2)$ operations per task, while the weight matrix update scales as $O(D \cdot C)$, where C is the number of classes. This complexity is significantly lower than approaches that require storing and processing exemplars from previous tasks.

Memory Efficiency: A key advantage of Algorithm 1 is its constant memory footprint with respect to the number of tasks. The algorithm maintains only the correlation matrix R_k and weight matrix $\hat{W}_{\text{FCN}}^{(k)}$, both of which have fixed sizes independent of the number of previously encountered tasks. This stands in stark contrast to replay-based methods, which require memory that grows linearly with the number of tasks.

Theoretical Foundation: The algorithm is grounded in the principle of analytic learning, which seeks to find closed-form solutions to optimization problems. By formulating the continual learning objective as a regularized least squares problem and employing the RFF approximation, the algorithm can compute exact solutions without iterative optimization. This theoretical foundation provides convergence guarantees and eliminates the hyperparameter tuning typically required for gradient-based methods.

The algorithm concludes by returning the final weight matrix $\hat{W}_{\text{FCN}}^{(K)}$, which encapsulates all learned knowledge from the sequence of tasks. This final model can then be used for inference on any class from any of the encountered tasks, effectively achieving the goal of continual learning without catastrophic forgetting for recursive updates throughout the learning process. The recursive nature of the updates, combined with the direct analytical solutions, enables the methodology to maintain performance across tasks without requiring access to historical data, making it particularly suitable for scenarios where data privacy and storage efficiency are paramount concerns.

Algorithm 1. ZeroRFF Training Algorithm

Require: Tasks $\mathcal{D}_1^{\text{train}}, \dots, \mathcal{D}_K^{\text{train}}$ with $\mathcal{D}_k^{\text{train}} \sim \{X_k^{\text{train}}, Y_k^{\text{train}}\}$; Pre-trained model with fixed weights Θ_{backbone}

Ensure: Weight matrix $\hat{W}_{\text{FCN}}^{(K)}$

1: Initialize: $R_0 \leftarrow \gamma I$, $\hat{W}_{\text{FCN}}^{(0)} \leftarrow 0$

2: **for** $k = 1$ to K **do**

3: $X_k^{(E)} \leftarrow f_{\text{backbone}}(X_k^{\text{train}}, \Theta_{\text{backbone}})$ ▷ Feature extraction

4: $X_k^{(B)} \leftarrow \text{rff}(X_k^{(E)})$ ▷ Apply Random Fourier Features

5: $R_k \leftarrow R_{k-1} - R_{k-1} X_k^{(B)T} (I + X_k^{(B)} R_{k-1} X_k^{(B)T})^{-1} X_k^{(B)} R_{k-1}$ ▷ Update correlation matrix

6: $\hat{W}_{\text{FCN}}^{(k)} \leftarrow [\hat{W}_{\text{FCN}}^{(k-1)} - R_{k-1} X_k^{(B)T} X_k^{(B)} \hat{W}_{\text{FCN}}^{(k-1)} + R_k X_k^{(B)T} Y_k^{\text{train}}]$ ▷ Update weight matrix

7: **end for**

4 Experiment and Results

4.1 Settings

We conducted experiments on a popular dataset in the fields of machine learning and computer vision: CIFAR-100 [11]. We evaluated under the Si-Blurry scenario [12], with 5 different seed values to ensure stability and reliability of the results. In the Si-Blurry setting, data is organized to reflect complex real-world scenarios where classes are not completely separated between tasks and there is label overlap. Specifically, the disjoint class ratio (*disjoint class ratio* (r_d)) was set to 50%, and the blurry sample ratio (*blurry sample ratio* (r_b)) was set at 10%.

4.2 Results

On the CIFAR-100 dataset, the proposed ZeroRFF method excels in the memory-free scenario, achieving $A_{AUC} = 60.48 \pm 4.33$, $A_{AVG} = 58.55 \pm 6.05$, và $A_{Last} = 72.44 \pm 0.13$. Compared to the second-best memory-free method, GACL ($A_{AUC} = 57.99 \pm 2.46$, $A_{AVG} = 56.24 \pm 3.12$, $A_{Last} = 70.31 \pm 0.06$), ZeroRFF improves by approximately 2% across all three criteria. Furthermore, the ZeroRFF method eliminates the dependence on memory storage while still outperforming most memory-based methods. Compared to MVP-R ($A_{AUC} = 60.62 \pm 1.03$ with memory size 2000), ZeroRFF is only slightly inferior in A_{AUC} , but superior in A_{AVG} and A_{Last} .

Table 1. Experimental results on CIFAR-100 (%).

Memory-size	Method	A_{AUC} (%)	A_{AVG} (%)	A_{Last} (%)
2000	EWC++ [13]	53.31 ± 1.70	50.95 ± 1.50	52.55 ± 0.71
	ER [14]	56.17 ± 1.84	53.80 ± 1.46	55.60 ± 0.69
	RM [15]	53.22 ± 1.82	52.99 ± 1.65	55.25 ± 0.61
	MVP-R [12]	60.62 ± 1.03	57.58 ± 0.56	64.30 ± 0.29
500	EWC++ [13]	48.31 ± 1.81	44.56 ± 0.40	40.52 ± 0.83
	ER [14]	51.59 ± 1.91	48.03 ± 0.80	44.09 ± 0.80
	RM [15]	41.07 ± 1.30	38.10 ± 0.59	32.66 ± 0.34
	MVP-R [12]	56.20 ± 1.44	53.61 ± 0.04	55.35 ± 0.43
0	LwF [16]	40.71 ± 1.21	38.49 ± 0.56	27.03 ± 2.92
	L2P [17]	42.68 ± 2.70	39.89 ± 0.45	28.59 ± 3.34
	DualPrompt [18]	41.34 ± 2.59	38.59 ± 0.82	22.74 ± 3.40
	MVP [12]	45.07 ± 1.24	44.93 ± 0.54	39.94 ± 0.47
	SLDA [19]	53.00 ± 3.85	50.09 ± 2.77	61.79 ± 3.81
	GACL [20]	57.99 ± 2.46	56.24 ± 3.12	70.31 ± 0.06
	ZeroRFF(Ours)	60.48 ± 4.33	58.55 ± 6.05	72.44 ± 0.13

5 Conclusion

This paper proposes ZeroRFF, an exemplar-free Generalized Class-Incremental Learning (GCIL) method aimed at addressing the catastrophic forgetting phenomenon and data privacy limitations. By integrating Analytic Continual Learning (ACL) techniques to directly solve for optimal solutions and Random Fourier Features (RFF) to transform data into a linearly separable space. Experimental results confirm that ZeroRFF outperforms other methods, providing an effective solution to address the continual learning problem.

References

1. French, R.M.: Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **3**(4), 128–135 (1999)
2. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint [arXiv:1312.6211](https://arxiv.org/abs/1312.6211) (2013)
3. Hou, S., Pan, X., Loy, C. C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839 (2019)
4. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
5. Mi, F., Kong, L., Lin, T., Yu, K., Faltings, B.: Generalized class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 240–241 (2020)
6. Rebuffi, S. A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: incremental classifier and representation learning. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010 (2017)
7. Guo, P., Lyu, M.R.: A pseudoinverse learning algorithm for feedforward neural networks with stacked generalization applications to software reliability growth data. *Neurocomputing* **56**, 101–121 (2004)
8. Zhuang, H., Weng, Z., Wei, H., Xie, R., Toh, K.A., Lin, Z.: Acil: analytic class-incremental learning with absolute memorization and privacy protection. *Adv. Neural. Inf. Process. Syst.* **35**, 11602–11614 (2022)
9. McDonnell, M.D., Gong, D., Parvaneh, A., Abbasnejad, E., Hengel, A.: Ranpac: random projections and pre-trained models for continual learning. *Adv. Neural. Inf. Process. Syst.* **36**, 12022–12053 (2023)
10. Zhuang, H., Weng, Z., He, R., Lin, Z., Zeng, Z.: Gkeal: Gaussian kernel embedded analytic learning for few-shot class incremental task. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7746–7755 (2023)
11. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
12. Moon, J.Y., Park, K.H., Kim, J.U., Park, G.M.: Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11731–11741 (2023)

13. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci.* **114**(13), 3521–3526 (2017)
14. Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., Wayne, G.: Experience replay for continual learning. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
15. Bang, J., Kim, H., Yoo, Y., Ha, J. W., Choi, J.: Rainbow memory: continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8218–8227 (2021)
16. Li, Z., Hoiem, D.: Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(12), 2935–2947 (2017)
17. Wang, Z., et al.: Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149 (2022)
18. Wang, Z., et al.: Dualprompt: complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 631–648. Springer, Cham, October 2022
19. Hayes, T.L., Kanan, C.: Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 220–221 (2020)
20. Zhuang, H., et al.: GACL: exemplar-free generalized analytic continual learning. *Adv. Neural. Inf. Process. Syst.* **37**, 83024–83047 (2024)



Towards Robust Continual Learning: A Multi-Head Approach with Online Prototype Equilibrium and Adaptive Prototypical Feedback

Quynh-Trang Pham Thi^(✉), Duc-Hung Nguyen, Thanh Hai Dang,
Duc-Trong Le, Tri-Thanh Nguyen, and Quang-Thuy Ha

VNU University of Engineering and Technology, Hanoi, Vietnam
{trangptq,22021109,hai.dang,trongld,ntthanh,thuyhq}@vnu.edu.vn

Abstract. Continual learning is an approach in machine learning that aims to learn different tasks sequentially, still performing well on all of them. This manner is akin to human learning. However, continual learning faces a significant challenge known as catastrophic forgetting. This refers to as a decrease in the model's performance on previously learned tasks when learning a new task. Memory-based replay method is one approach that has been proven effective in addressing this issue. After completing each task, the model stores a small amount of data to combine with new data for training an encountering task. This approach is associated with the size of the memory buffer and data security concerns. In this paper, instead of storing actual data points, we only store prototypes representing classes in learned tasks. To this end, we also employ two techniques, i.e. Online Prototype Equilibrium (OPE) and Adaptive Prototypical Feedback (APF) to enhance the quality of prototypes' hidden representation. Furthermore, to enhance accurate classification, we do not use a single shared head for all classes. Instead, for each task, we add a within-task prediction head and a task-ID prediction head. Experimental results on benchmark datasets demonstrate that our method outperforms several state-of-the-art methods in terms of well-studied average accuracy.

Keywords: continual learning · online prototype · catastrophic forgetting · class incremental learning

1 Introduction

Traditional machine learning models often assume that data from all tasks were simultaneously seen, which does not hold true in reality. To mimic the human's learning capacity, continual learning (CL) models must be sequentially trained on each task. Nevertheless, when learning a new task, models need to update their previously learned parameters to optimize for new incoming data. This update

can alter the parameters important for the previous tasks, diminishing models' performance on learned tasks. This phenomenon is called catastrophic forgetting - a major challenge of continual learning (CF) [1]. Early studies in CF focus on the task-incremental learning (TIL) scenario, in which the model chooses the task-specific component for classification based on the task identity [2, 3, 11]. However, TIL setting lacks flexibility in real-world scenarios. In this paper, we focus on a more general and realistic setting, namely the class-incremental learning (CIL) in the online CL mode [4–6], in which the model learns incrementally classes in a sequence of tasks from a single-pass data stream and cannot access task identifiers at the inference.

Rehearsal-based strategies, involving the storage and replay of past task samples (often impossible in reality), have demonstrated considerable successes in mitigating CF [7]. Furthermore, storing prior task data often results in an expanding memory requirement as the number of tasks increases [8, 9], which can be impractical under strict memory constraints. As a result, the development of methods capable of matching or surpassing the efficiency of data-storage-based approaches is highly crucial.

To deal with this challenge, instead of storing actual data points, class prototypes as of vector representations [11] can be in use. Predictions for a target class can be made based on the similarity between these prototypes and new samples' representations. The efficiency of a continual learner can be enhanced by accurately maintaining estimates of previous class prototypes [12]. The overlap between classes is an indication of catastrophic forgetting. To address this class confusion, numerous studies [15–17] utilize self-distillation [13, 14] to retain prior knowledge. Nevertheless, knowledge distillation requires that adequate discriminative features have been acquired for former classes, and these features continue to be discriminative when learning a new class.

The contributions of this work are two-fold. Firstly, we propose to employ two techniques, i.e., Online Prototype Equilibrium (OPE) and Adaptive Prototypical Feedback (APF), to enhance the quality of hidden features. Secondly, to improve the model's classification performance, for each task, we propose to use: (i) the within-task prediction head to teach the our CF model how to recognize data belonging to that task and (ii) the out-of-distribution head to teach the model how to identify data belonging to a specific class within that task.

2 Related Work

Apart from CF, CIL presents another intricate challenge, i.e. the inter-task class separation (ICS). ICS occurs when the learner lacks access to the training dataset of former tasks when learning a new task. Recent studies underpin three conditions necessary and sufficient for a robust and effective CIL model. They are (i) proficient within-task prediction (WP), (ii) proficient task-id prediction (TP) and (iii) excellent out-of-distribution (OOD) detection for each task.

For a training set $D = \{(x^i, y^i)_{i=1}^n\}$ of each task, where n is the number of data samples, $x^i \in \chi$ is an input sample and $y^i \in \gamma$ is its corresponding label. γ is

the set of all labels in D and is referred to as the in-distribution (IND). The goal of OOD detection is to construct a function $f : \chi \rightarrow \gamma \cup \{O\}$ that can identify Out-of-Distribution (OOD) instances not belonging to any class in γ . The class of such instances is denoted as O . When the model can effectively recognize OOD instances for each task, it hence addresses the inter-task class separation (ICS) problem [19]. ICS refers to the situation in which the learner encounters challenges in establishing clear boundaries between classes of different tasks.

Regarding WP and TP, Kim et al. [18,19], present a formula for calculating the probability of a test sample belong to a task, which is the product of two probabilities, i.e. WP and TP as in formula 1 and 2 as belows, respectively.

$$\mathbf{P}_{WP} = \mathbf{P}(x \in X_{k,j} | x \in \mathbf{X}_k, D), \quad (1)$$

$$\mathbf{P}_{TP} = \mathbf{P}(x \in \mathbf{X}_k | D), \quad (2)$$

where $\mathbf{X}_k$ is the dataset of task k , $\mathbf{Y}_k$ is the set of classes for task k and $\mathbf{X}_k = \{\mathbf{X}_{k,j}\}$, j represents the j^{th} class of the k^{th} task. The classes within each task are distinct, i.e., $\mathbf{X}_{k,j} \cap \mathbf{X}_{k,j'} = \emptyset, \forall j \neq j'$. The tasks are also distinct, i.e., $\mathbf{X}_k \neq \mathbf{X}_{k'} = \emptyset, \forall k \neq k'$. In this line, the prediction probability of a CIL model is given by formula 3 as follows:

$$\begin{aligned} \mathbf{P}(x \in \mathbf{X}_{k_0,j_0} | D) &= \sum_{k=1, \dots, n} \mathbf{P}(x \in X_{k,j_0} | x \in \mathbf{X}_k, D) \mathbf{P}(x \in \mathbf{X}_k | D) \\ &= \mathbf{P}(x \in X_{k_0,j_0} | x \in \mathbf{X}_{k_0}, D) \mathbf{P}(x \in \mathbf{X}_{k_0} | D), \end{aligned} \quad (3)$$

Kim et al., 2023 proposed a continual learning method implemented with a replay-based Class Incremental Learning called ROW [18] (ROW is an abbreviation for Replay, OOD, WP). At k^{th} task, the system receives the dataset D_k and leverages the replayed data stored from previous tasks in the memory buffer M as out-of-distribution data for task k to train an OOD detection head while adjusting a WP head simultaneously. Specifically, the model for each task consists of two heads: one head for OOD to compute TP and another head for WP. The model trains and fine-tunes the parameters ψ_k of the feature extraction function f_k , the parameters θ_k of the OOD head h_k , and the parameters ϕ_k of the WP head similar to a classification head g_k . Both h_k and g_k heads receive the same features extracted by f_k to calculate the corresponding probabilities.

The training process consists of 3 phases: (1) training the feature extractor f_k and the OOD head h_k by using the IND dataset in D_k and the OOD dataset in the buffer; (2) fine-tuning the WP head g_k for the task using data D_k solely based on the fixed feature extractor f_k ; and (3) fine-tuning the OOD heads of all previously learned tasks. Training phases (2) and (3) involve simply fine-tuning a single layer of the classifier. The outputs of two heads are used to compute the final CIL prediction probability in formula 3.

3 Method

In this study, we aim to optimize the quality of hidden feature representations through Online Prototype Equilibrium(OPE) and Adaptive Prototypical Feed-

back (APF). To enhance the model’s classification capability, during the training of each task, we incorporate two heads: the WP head (to improve task recognition) and the OOD head (to enhance class recognition).

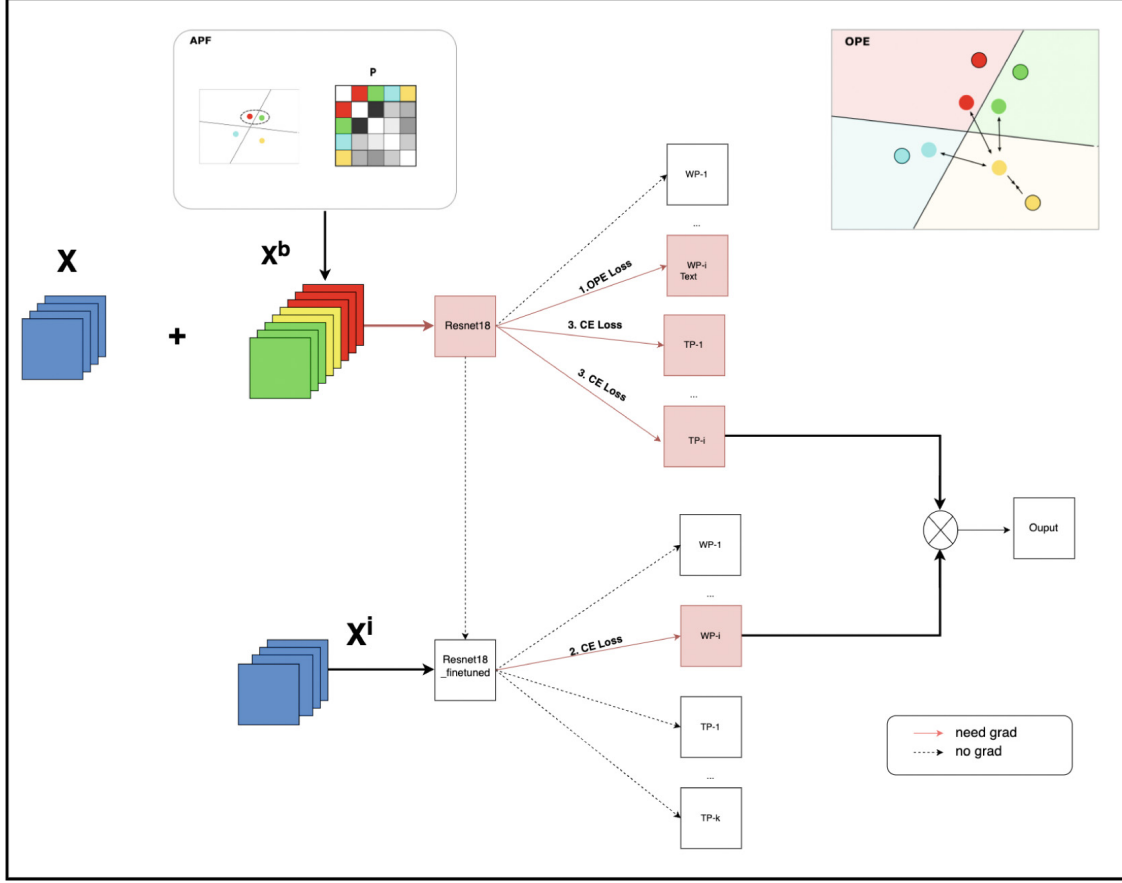


Fig. 1. The overall architecture of our proposed method.

3.1 Feature Representation Optimization

OPE learns the representative features for each class by **pulling** online prototypes both from the data batch $\mathcal{P}$ and the augmented data batch $\hat{\mathcal{P}}$ that are close to each other in the embedding space (depicted by two arrows pointing towards each other in Fig. 1). OPE enhances the discriminative boundaries between classes by **pushing** online prototypes from other classes away from $\mathcal{P}$ and $\hat{\mathcal{P}}$ in the embedding space (depicted by the bidirectional arrows in Fig. 1).

In each epoch of task t , the data received in small batches is denoted as X . Our proposed method operates on the entire data batch instead of processing each data sample. Therefore, the **online prototype** of each class in an epoch is defined as the averaged representation of all online prototypes of the input data within a small batch:

$$\mathbf{p}_i = \frac{1}{n_i} \sum_j \mathbf{z}_j \cdot \mathbb{1}\{y_j = i\}, \quad (4)$$

where n_i is the number of samples of class i in a small batch, and $\mathbb{1}$ is an indicator function. A set K of online prototypes in X is denoted as: $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^K$ ($K = |\mathcal{P}| \leq |\mathcal{C}_t|$), similar to a set of online prototypes $\mathcal{K}^b$ in X^b , $\mathcal{P}^b = \{\mathbf{p}_i^b\}_{i=1}^{K^b}$ ($K = |\mathcal{P}| \leq |\mathcal{C}_t|$ and $K^b = |\mathcal{P}^b| \leq \sum_{i=1}^t |\mathcal{C}_i|$) with $\mathcal{C}$ being the set of classes in the data batch.

OPE adjusts the network parameters to optimize the two aforementioned training phases by minimizing the contrastive loss function given by the formula:

$$l(\mathcal{P}, \hat{\mathcal{P}}) = \frac{-1}{|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} \log \frac{\exp(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_i}{\tau})}{\sum_j \exp(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_j}{\tau}) + \sum_{j \neq i} \exp(\frac{\mathbf{p}_i^T \mathbf{p}_j}{\tau})}, \quad (5)$$

where τ is the temperature hyperparameter, and $\mathcal{P}$ and $\hat{\mathcal{P}}$ are l_2 normalized. The contrastive loss across all positive pairs in both $(\mathcal{P}, \hat{\mathcal{P}})$ and $(\hat{\mathcal{P}}, \mathcal{P})$ is given by the formula:

$$\mathcal{L}_{\text{pro}}(\mathcal{P}, \hat{\mathcal{P}}) = \frac{1}{2} [l(\mathcal{P}, \hat{\mathcal{P}}), l(\hat{\mathcal{P}}, \mathcal{P})], \quad (6)$$

To this end, OPE adjusts the model parameters to effectively learn both discriminative and distinctive features for both new and previously learned classes using $\mathcal{L}_{\text{pro}}$, given by the formula:

$$\mathcal{L}_{\text{OPE}} = \mathcal{L}_{\text{pro}}^{\text{new}}(\mathcal{P}, \hat{\mathcal{P}}) + \mathcal{L}_{\text{pro}}^{\text{seen}}(\mathcal{P}^b, \hat{\mathcal{P}}^b), \quad (7)$$

where $\mathcal{L}_{\text{pro}}^{\text{new}}$ focuses on learning knowledge from new classes, and $\mathcal{L}_{\text{pro}}^{\text{seen}}$ is dedicated to preserving the knowledge of all previously known classes. *This process is akin to a zero-sum game, and OPE aims to achieve a balance to play a mutually beneficial game.* Specifically, as the model learns, the knowledge from a new class is gathered and added to prototypes in the memory bank $\mathcal{M}$, causing $\mathcal{L}_{\text{pro}}^{\text{new}}$ to gradually shift towards a balanced state that separates all known classes from each other better, including the new one. This adaptation is crucial for minimizing CF and aligns with the goals of Class Incremental Learning.

While OPE can achieve a global balance, it tends to treat each class uniformly. In practice, varying degrees of confusion exist between classes. As a result, the model should prioritize classes more susceptible to confusion. We propose to exert APF to incorporate feedback from online prototypes for identifying classes prone to misclassification, thus enhancing their decision boundaries. For each class pair in the memory bank $\mathcal{M}$, APF computes the distance between online prototypes of all previously known classes, indicating the confusion status based on these distances. The closer two prototypes are, the more prone they are to misclassification. Therefore, APF transforms the prototype distance matrix into a probability distribution $\mathbf{P}$ over classes through a symmetric Gaussian kernel, defined as follows:

$$P_{i,j} \propto \exp(-\|\mathbf{p}_i^b - \mathbf{p}_j^b\|_2^2) \quad (8)$$

where $i, j \in 1, \dots, |\mathcal{P}^b|$ and $i \neq j$. Subsequently, all the probabilities are normalized into a probability mass function summing to 1. APF returns probabilities

for $\mathcal{M}$ to guide the next sampling process and enhances the decision boundaries of classes prone to misclassification.

APF is implemented as a sampling-based ensemble. Specifically, APF selectively samples more instances from classes prone to misclassification within $\mathcal{M}$ to mix according to the probability distribution P . Considering the potential over-penalization of the current balanced state of online prototypes, we employ a two-stage sampling strategy for the replay data X^b of size m . Firstly, select n_{APF} samples using P , with larger $P_{a,b}$ meaning more samples are drawn from classes a and b . Here, $n_{\text{APF}} = \alpha.m$, and α is the ratio controlled by APF. Secondly, the remaining $m - n_{\text{APF}}$ samples are randomly chosen from the entire memory bank to prevent the model from solely focusing on classes prone to misclassification, disrupting the established balanced state. An overview of the APF computation technique is outlined in Algorithm 1.

Algorithm 1: Algorithm for the APF technique [10]

Input: $\mathcal{M}$ and the set of online prototypes $\{\mathbf{p}_i\}_{i=1}^{K^b}$ from previous time steps.
Output: X^b
Initialization: $\mathcal{S} \leftarrow \{\}$, $n_{\text{APF}} = \alpha.m$, $P \leftarrow$ compute probabilities $P_{i,j}$ for each class pair using $\mathbf{p}_i^b$ và $\mathbf{p}_j^b$

- 1 **for** *For each* $P_{i,j} \in P$ **do**
- 2 $X_i, X_j \leftarrow [P_{i,j}.n_{\text{APF}} + 0.5]$ the number of samples from class i and class j
- 3 $\mathcal{S} \leftarrow \mathcal{S} \cup \text{Mixup}(X_i, X_j)$
- 4 **end**
- 5 $X_{\text{base}} \leftarrow$ the remaining $m - n_{\text{APF}}$ samples randomly selected from $\mathcal{M}$
- 6 $X^b \leftarrow \mathcal{S} \cup \text{Mixup}(X_{\text{base}}, X_{\text{base}})$

3.2 Classification Layer Optimization

Inspired by [18], especially with the following formula:

$$\mathbf{P}(X_{k,j}|x) = \mathbf{P}(X_{k,j}|x, k)\mathbf{P}(X_k|x), \quad (9)$$

We propose to add the OOD Head (for computing TP) and WP Head (for computing WP) at the classifier layer to enhance our model's performance. Our proposed model optimizes the parameter set $(\Psi_k, \theta_k, \phi_k)$, where Ψ_k is the parameters of OOD head h_k and ϕ_k are the parameters of WP head. In this line, the training process consists of three steps:

1. Train the feature extractor f_k and the OOD head h_k using the samples from IND in D_k and OOD samples in $\mathcal{M}$.
2. Fine-tune the WP head g_k for the current task using D_k based on the fixed feature extractor f_k .
3. Fine-tune the OOD head for all previous tasks. In steps 2 and 3, only the last layer is fine-tuned, while the feature extractor remains fixed.

Training Feature Extractor and OOD Head: This step trains the OOD head h_k for task k . Its feature extractor f_k is also shared by the WP head (see below). An illustration of the training process is given in Fig. 1(a). Since the OOD instances are any instances whose labels do not belong to task k , we leverage the task-specific data D_k as IND instances and the saved replay instances of tasks $k' \neq k$ in the memory buffer $\mathcal{M}$ as OOD instances of an OOD class (in red) in the OOD head. The network $h_k \circ f_k$ is trained to maximize the probability $p(y|x, k) = \text{softmax}_{h_k}(f(x, k; \Psi_k); \theta_k)_y$ for an IND instance $x \in D_k$ and maximize the probability $p(\text{ood}|x, k)$ for OOD instance $x \in \mathcal{M}$. Formally, this is achieved by minimizing the sum of cross-entropy losses as in formula 10

$$\mathcal{L}_{\text{ood}}(\Psi_t, \theta_t) = -\frac{1}{2N} \left(\sum_{(x,y) \in D_k} \log p(y|x, k) + \sum_{(x,y) \in \mathcal{M}} \log p(\text{ood}|x, k) \right), \quad (10)$$

where N is the number of instances in D_k . We utilize upsampling with the replay instances to achieve an equal number of samples as the current task data D_k . The first loss discriminates IND instances while the second is introduced to distinguish between IND and OOD instances.

Fine-Tuning the WP Head: Given the feature extractor trained in the first step, we fix the feature extractor and fine-tune the WP head g_k (i.e., the WP classifier) using only D_k by adjusting the parameters ϕ_k . This is achieved by minimizing the cross-entropy loss as follows.

$$\mathcal{L}_{WP}(\phi_k) = -\frac{1}{N} \sum_{(x,y) \in D_k} \log p(y|x, k). \quad (11)$$

WP probabilities for the classes of task k are just the output softmax probabilities.

Fine-Tuning the OOD Heads of All Tasks: The OOD head h_k built in step 1) is biased because for early tasks, where the instances in $\mathcal{M}$ are less diverse, the OOD heads for them will be weaker than the OOD heads for later tasks when instances in $\mathcal{M}$ are more diverse. To mitigate this bias, we propose to fine-tune all OOD heads of all tasks after training each task using only the replay data in $\mathcal{M}$. After training task k , we have $\mathcal{M}$ with replay instances of classes from task 1 to k . For each task $k' \leq k$, reconstruct a new IND data $\tilde{D}_{k'}$ by selecting instances corresponding to task k' from $\mathcal{M}$, and a new pseudo memory buffer $\tilde{\mathcal{M}}$ after removing the instances in $\tilde{D}_{k'}$. We then fine-tune every OOD head by minimizing the loss function as in formula 12.

$$\mathcal{L}_{TP}(\theta_{k'}) = -\frac{1}{M} \left(\sum_{(x,y) \in \tilde{\mathcal{M}}} \log p(\text{ood}|x, k') + \sum_{(x,y) \in \tilde{D}_{k'}} \log p(y|x, k') \right), \quad (12)$$

where M is $|\tilde{D}_{k'}| + |\tilde{\mathcal{M}}|$.

4 Experiments and Results

4.1 Evaluation Metrics

We employ two benchmark datasets for image classification, namely CIFAR-10 and CIFAR-100 to assess the effectiveness of our proposed model that is based on the online Class Incremental Learning (CIL) approach. We utilize two metrics to measure the performance of our method: average accuracy and average forgetting.

Average Accuracy evaluates the average accuracy of the model across all tasks:

$$ACC = \frac{1}{T} \sum_{j=1}^T a_{T,j}, \quad (13)$$

Where $a_{T,j}$ is the accuracy of task j after the model has been trained on all tasks.

Average Forgetting represents the extent to which the model forgets each task after being trained on the final task and is calculated using the formula:

$$AF = \frac{1}{T-1} \sum_{j=1}^{T-1} f_{T,j}, \quad (14)$$

where $f_{T,j} = \max_{k \in \{1, \dots, i-1\}} a_{k,j} - a_{i,j}$.

4.2 Results

Our model achieves the highest effectiveness compared to several other baseline methods in terms of the average accuracy metric, as demonstrated in Table 1. When using our proposed multi-head method, the model outperforms the Onpro method that uses a single head, with experiments on the Cifar-10 dataset showing an increase of nearly 9% from 52.4% to 63.62% when $\mathcal{M} = 100$, and a 6% increase when $\mathcal{M} = 200$, and 5% when $\mathcal{M} = 500$. The experimental results on the Cifar-100 dataset also show significant improvements, increasing by approximately 7%, 3%, and 1.3% respectively for $\mathcal{M} = 500$, $\mathcal{M} = 1000$, and $\mathcal{M} = 2000$. This indicates that our method improves significantly more when using a smaller memory buffer size.

The results in Table 2 indicate that, despite achieving higher average accuracy, our proposed model still faces the issue of catastrophic forgetting. For example, in experiments on CIFAR-10, with $\mathcal{M} = 100$ and $\mathcal{M} = 500$, our model has the smallest forgetting scores, but with $\mathcal{M} = 200$, it is larger compared to Onpro (Note that a smaller forgetting score is better). For the CIFAR-100 dataset, our model only forgets the least with $\mathcal{M} = 1000$, while in the other two cases, it is higher than Onpro. This is explained by the fact that our method is currently focusing on optimizing for classification ability of the model and has not addressed the issue of catastrophic forgetting as much.

Table 1. Average accuracy of our model and SOTA models with different buffer sizes $\mathcal{M}$

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL [20]	31	33.9	42	12.8	16.5	17.6
DER++ [21]	31.5	39.7	50.9	16	21.4	23.9
PASS [22]	33.7	33.7	33.7	7.5	7.5	7.5
AGEM [8]	17.7	17.5	17.5	5.8	5.9	5.8
GSS [23]	18.4	19.4	25.2	8.1	9.4	10.1
ER [24]	19.4	20.9	26	8.7	9.9	10.7
MIR [25]	20.7	23.5	29.9	9.7	11.2	13
GDumb [26]	23.3	27.1	34	8.2	11	15.3
ASER [27]	20	22.8	31.6	11	13.5	17.6
SCR [28]	40.2	48.5	59.1	19.3	26.5	32.7
CoPE [11]	33.5	37.3	42.9	11.6	14.6	16.8
DVC [29]	35.2	41.6	53.8	15.4	20.3	25.2
OCM [15]	47.5	59.6	70.1	19.7	27.5	34.4
OnPro [10]	52.4	63	67.7	20.7	27.2	34.9
Ours	63.62	69.34	72.78	27.06	30.29	36.28

Table 2. Average Forgetting of our model and SOTA models with different buffer sizes $\mathcal{M}$.

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL [20]	52.7	49.3	38.3	16.5	11.2	10.4
DER++ [21]	57.8	46.7	33.6	41.0	34.8	33.2
PASS [22]	21.2	21.2	21.2	10.6	10.6	10.6
AGEM [8]	64.8	64.8	64.5	41.7	41.8	41.7
GSS [23]	67.1	65.8	61.2	48.7	46.7	44.7
ER [24]	64.7	62.9	57.5	47.0	46.4	44.7
MIR [25]	62.6	58.5	51.1	45.7	44.2	42.3
GDumb [26]	28.5	28.4	28.1	25.0	23.2	20.7
ASER [27]	64.8	62.6	53.2	52.8	50.4	46.8
SCR [28]	43.2	35.5	24.1	29.3	20.4	11.5
CoPE [11]	49.7	45.7	39.4	25.6	17.8	14.4
DVC [29]	40.2	31.4	21.2	32.0	32.7	28.0
OCM [15]	35.5	23.9	13.5	18.3	15.2	10.8
OnPro [10]	24.2	16.5	13.7	17.8	14.9	10.0
ModifiedOnPro	23.2	18.25	13.5	19.24	14.47	11.19

We track task accuracy metrics during the training process, illustrating that our method consistently outperforms the Onpro method with higher results (see Fig. 2).

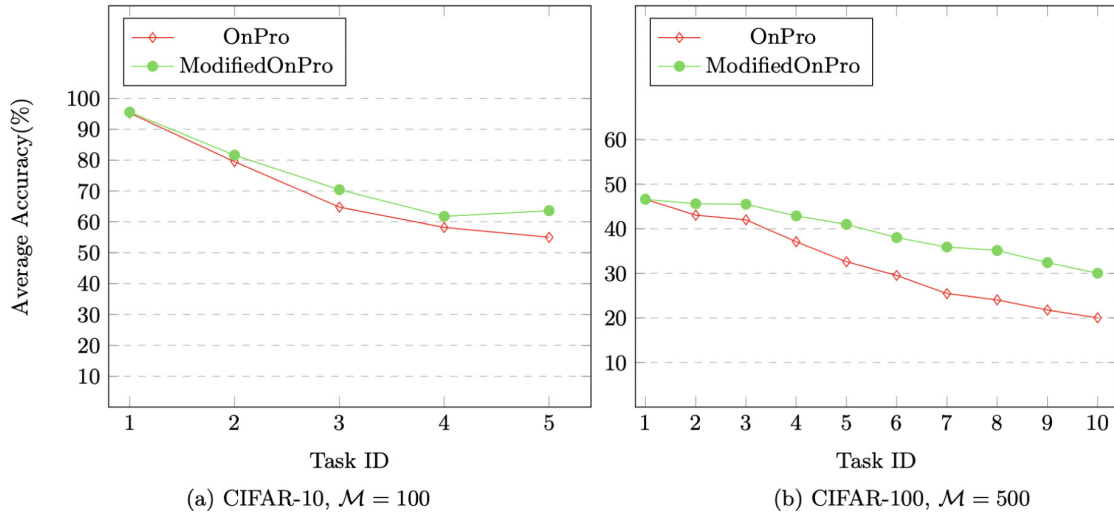


Fig. 2. Task accuracy metrics during the training process of our method (green line) vs. the Onpro method (red line). (Color figure online)

5 Conclusion

In this paper, we study continual learning in the Class Incremental Learning setting. To tackle the big challenge of catastrophic forgetting in continual learning, we propose to optimize both feature extraction and classification. In the feature extraction step, we propose to employ two techniques, i.e. Online Prototype Equilibrium (OPE) and Adaptive Prototypical Feedback (APF). OPE endeavours to bring the representations of samples from the same class closer while pushing representations of samples from different classes farther apart. In the classification layer, instead of using a single head, we propose to utilize a multi-head classifier. For each task, we have two heads: the WP head to enhance the ability to identify which task the input image belongs to, and the OOD head to improve the ability to recognize which class the input image belongs to. Experimental results demonstrate the effectiveness of our proposed method, outperforming existing state-of-the-art models.

References

1. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. In: *Psychology of Learning and Motivation*, vol. 24, pp. 109–165. Academic Press (1989)
2. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: learning what (not) to forget. In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 139–154 (2018)
3. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming catastrophic forgetting with hard attention to the task. In: *International Conference on Machine Learning*, pp. 4548–4557. PMLR, July 2018
4. Mai, Z., Li, R., Jeong, J., Quispe, D., Kim, H., Sanner, S.: Online continual learning in image classification: an empirical survey. *Neurocomputing* **469**, 28–51 (2022)

5. Chrysakis, A., Moens, M.F.: Online continual learning from imbalanced data. In: International Conference on Machine Learning, pp. 1952–1961. PMLR, November 2020
6. He, J., Mao, R., Shao, Z., Zhu, F.: Incremental learning in online scenario. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13926–13935 (2020)
7. Asadi, N., Davari, M., Mudur, S., Aljundi, R., Belilovsky, E.: Prototype-sample relation distillation: towards replay-free continual learning. In: International Conference on Machine Learning, pp. 1093–1106. PMLR, July 2023
8. Chaudhry, A., Ranzato, M.A., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. arXiv preprint [arXiv:1812.00420](https://arxiv.org/abs/1812.00420) (2018)
9. Caccia, M., et al.: Online fast adaptation and knowledge accumulation: a new approach to continual learning. arXiv preprint [arXiv:2003.05856](https://arxiv.org/abs/2003.05856) (2020)
10. Wei, Y., Ye, J., Huang, Z., Zhang, J., Shan, H.: Online prototype learning for online continual learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 18764–18774 (2023)
11. De Lange, M., Tuytelaars, T.: Continual prototype evolution: learning online from non-stationary data streams. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8250–8259 (2021)
12. Davari, M., Asadi, N., Mudur, S., Aljundi, R., Belilovsky, E.: Probing representation forgetting in supervised and unsupervised continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16712–16721 (2022)
13. Fang, Z., Wang, J., Wang, L., Zhang, L., Yang, Y., Liu, Z.: Seed: Self-supervised distillation for visual representation. arXiv preprint [arXiv:2101.04731](https://arxiv.org/abs/2101.04731) (2021)
14. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint [arXiv:1503.02531](https://arxiv.org/abs/1503.02531) (2015)
15. Guo, Y., Liu, B., Zhao, D.: Online continual learning through mutual information maximization. In: International Conference on Machine Learning, pp. 8109–8126. PMLR, June 2022
16. Li, Z., Hoiem, D.: Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(12), 2935–2947 (2017)
17. Cha, H., Lee, J., Shin, J.: Co2l: contrastive continual learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9516–9525 (2021)
18. Kim, G., Xiao, C., Konishi, T., Liu, B.: Learnability and Algorithm for Continual Learning. arXiv preprint [arXiv:2306.12646](https://arxiv.org/abs/2306.12646) (2023)
19. Kim, G., Xiao, C., Konishi, T., Ke, Z., Liu, B.: A theoretical study on solving continual learning. *Adv. Neural. Inf. Process. Syst.* **35**, 5065–5079 (2022)
20. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2001–2010 (2017)
21. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. *Adv. Neural. Inf. Process. Syst.* **33**, 15920–15930 (2020)
22. Zhu, F., Zhang, X.Y., Wang, C., Yin, F., Liu, C.L.: Prototype augmentation and self-supervision for incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5871–5880 (2021)
23. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32 (2019)

24. Chaudhry, A., et al.: On tiny episodic memories in continual learning. arXiv preprint [arXiv:1902.10486](https://arxiv.org/abs/1902.10486) (2019)
25. Aljundi, R., et al.: Online continual learning with maximal interfered retrieval. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019). 1, 2, 5, 6, 7
26. Prabhu, A., Torr, P.H.S., Dokania, P.K.: GDumb: a simple approach that questions our progress in continual learning. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) *ECCV 2020. LNCS*, vol. 12347, pp. 524–540. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_31
27. Shim, D., Mai, Z., Jeong, J., Sanner, S., Kim, H., Jang, J.: Online class-incremental continual learning with adversarial shapley value. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, No. 11, pp. 9630–9638, May 2021
28. Mai, Z., Li, R., Kim, H., Sanner, S.: Supervised contrastive replay: revisiting the nearest class mean classifier in online class-incremental continual learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3589–3599 (2021)
29. Gu, Y., Yang, X., Wei, K., Deng, C.: Not just selection, but exploration: online class-incremental continual learning via dual view consistency. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7442–7451 (2022)

Vietnam Journal of Computer Science
© World Scientific Publishing Company

Addressing Catastrophic Forgetting in Continual Learning using Multihead Approach and Latent Alignment via Energy-Based Model

Quynh-Trang Pham Thi, Duc-Hung Nguyen, Anh-Duc Nguyen-Tran, Hung-Dung Nguyen, Tri-Thanh Nguyen, Quang-Thuy Ha and Thanh Hai Dang*

*Faculty of Information Technology,
VNU University of Engineering and Technology,
Hanoi, Vietnam*

*{trangptq, 22021109, 23025059, 22021133, ntthanh, thuyhq,
hai.dang}@vnu.edu.vn*

Received (Day Month Year)

Revised (Day Month Year)

When incrementally trained for new tasks, deep continual learning models typically suffer from catastrophic forgetting, where parameter updates optimized for new tasks deteriorate performance on previously mastered tasks. This paper proposes a comprehensive replay-based approach built upon the OnPro model to address this fundamental challenge through three synergistic contributions. First, we employ Equilibrium Training with Prototypical Feedback to enhance discriminative feature quality through adaptive prototype evolution. Second, we introduce a specialized multi-head architecture incorporating (i) within-task prediction (WP) heads for task-specific classification and (ii) out-of-distribution (OOD) heads for inter-task class separation. Third, we develop an Energy-based Latent aligner (ELI) that learns energy manifolds to minimize representational drift, assigning higher energy to current task representations and lower energy to previous task features. Our method demonstrates substantial improvements on medical imaging (CCH5000) and standard benchmarks (CIFAR-10/100), achieving 27.5% forgetting reduction on medical datasets and up to 21.4% accuracy improvement on CIFAR-10 under memory constraints. However, our analysis reveals important trade-offs: computational overhead increases by 82%, and effectiveness diminishes in single-class scenarios due to insufficient negative sampling diversity for contrastive learning. These findings provide both methodological advances and theoretical insights that guide future continual learning research.

Keywords: continual learning; continual image classification; energy-base model; medical image classification.

1. Introduction

Traditional machine learning models heavily rely on the availability of complete training datasets for learning those models, an assumption that proves not true in many practical scenarios^{1,2}. Unlike these static learning paradigms, humans demonstrate exceptional capacity for lifelong knowledge acquisition across multiple tasks, seamlessly integrating new

*Corresponding author

This paper is a corrected and extended version of our ACIIDS 2024 conference paper Pham Thi et al., 2024.

information from new tasks while preserving existing capabilities for old ones. This remarkable ability has inspired researchers to develop models capable of continuous learning across sequential tasks³. Nevertheless, machine learning models, especially neural networks, face a fundamental challenge when exposed to evolving data distributions from new tasks: optimizing parameters for new tasks' objectives often leads to severe performance degradation on previously mastered tasks^{4,5}. This issue, termed catastrophic forgetting (CF), emerges because gradient-based updates modify weights essential for earlier learned tasks, effectively overwriting stored knowledge⁶.

Continual learning research has established three primary learning configurations that are distinguished by their assumptions regarding data availability and task patterns⁷. Task-incremental learning (TIL) assumes the availability of explicit task identifiers throughout training and inference phases. Domain-incremental learning (DIL) settings involve adaptation across varying input distributions while preserving consistent output spaces. Class-incremental learning (CIL) configurations present the greatest challenge, demanding recognition of novel classes without knowing task identifiers during inference⁸. Class-incremental learning most accurately reflects realistic learning scenarios where new classes emerge naturally within continuous data streams without predetermined transition markers⁹. This configuration demands sophisticated mechanisms for discriminating classes from different learning episodes while preserving performances for all encountered classes.

Consequently, apart from catastrophic forgetting, CIL presents another intricate challenge, i.e. the inter-task class separation (ICS). ICS occurs when the learner lacks access to the training dataset of former tasks when learning a new task. Recent studies underpin three conditions necessary and sufficient for a robust and effective CIL model. They are (i) proficient within-task prediction (WP), (ii) proficient task-id prediction (TP) and (iii) excellent out-of-distribution (OOD) detection for each task.

Research efforts to combat catastrophic forgetting have fallen into three distinct methodological categories¹⁰: (i) **Regularization-based approaches** that impose constraints preventing modifications to network parameters critical for previous task performance^{6,11,12}; (2) **Architecture-based methods** that expand or reconfigure network structures to create dedicated capacity for new knowledge^{13,14,15}; and (3) **Replay-based techniques** that maintain explicit stores of previous task examples that are revisited during subsequent learning^{16,17,8,18}. Replay based methodologies have achieved particularly impressive empirical outcomes^{19,20}, yet they encounter significant practical obstacles: memory demands increase proportionally with the number of tasks²¹, and explicit data retention may expose privacy concerns in sensitive domains²². These limitations call for more memory-efficient alternatives.

Early studies in continual learning focused primarily on task-incremental learning (TIL) scenarios where task identifiers were explicitly provided, enabling models to select task-specific appropriate components for inference^{23,2,24,23}. Although this simplified setting enables controlled experimentation, it imposes unrealistic assumptions for real-world applications where task identifiers are unknown⁷.

In this work, we tackle the substantially more challenging class-incremental learning (CIL) paradigm, where models must identify and learn new categories as they emerge in

data streams, without advance knowing task identifiers^{9,8}. To succeed in this scenario, models must simultaneously: (a) rapidly acquire knowledge about novel classes, (b) preserve discrimination capabilities for all previously encountered classes, and (c) make appropriate predictions without task identifiers.

Our replay-based approach builds upon prototype-based learning paradigms²⁵, representing classes through learnable compact vector embeddings rather than explicit raw data storage². This strategy offers computational efficiency while maintaining representational flexibility. However, effective implementation requires careful orchestration of the evolution of prototypes and the stability of feature space across learning episodes. To address this challenge, we introduce three complementary contributions, as follows.

- First, we propose employing Equilibrium Training with Prototypical Feedback to enhance the quality of hidden features.
- Second, to improve the model’s classification performance, for each task, we propose to use: (i) the within-task prediction (WP) head to guide our model how to recognize data belonging to that task and (ii) the out-of-distribution (OOD) head to guide the model how to identify data belonging to a specific class within that task.
- Third, we further improve OnPro by applying an energy-based model from²⁶ to minimize the representational shift in the latent space (in hidden layers) of the incremental model. We learn an energy manifold where the latent representations from the model trained until the current task have higher energy, while those from the model trained until the previous task have lower energy. This learned energy-based model helps prevent representation shifts across learning episodes.

The first two contributions are described in our ACIIDS 2024 conference paper²⁷, from which this paper extends with the third conclusion. Comprehensive evaluation on various medical imaging datasets and established computer vision benchmarks reveals significant performance improvements compared to existing approaches, with especially pronounced benefits under memory-limited conditions representative of practical deployment scenarios.

2. Related Work

2.1. Catastrophic Forgetting Intimidation

Regularization-based continual learning prevents catastrophic forgetting through parameter update constraints that protect knowledge-critical network weights^{6,11}. Elastic Weight Consolidation⁶ introduces quadratic penalty terms derived from Fisher Information estimates, discouraging modifications to parameters essential for previous task performance. Synaptic Intelligence¹¹ adopts an alternative strategy that computes parameter contributions online throughout training, providing computational benefits over EWC’s offline Fisher Information calculation. Recent theoretical developments¹² have established statistical frameworks for understanding convergence characteristics of regularization methods in analytically tractable scenarios.

Architecture-based solutions prevent parameter interference through dedicated parameter allocation for individual tasks^{13,14,15}. Progressive Neural Networks¹³ construct separate network columns for each task while enabling knowledge sharing via cross-connections between them. This design ensures complete forgetting avoidance but requires linear parameter growth with task quantity. Expert Gate¹⁵ addresses scalability challenges through learned input routing that directs samples toward appropriate specialized networks. PackNet¹⁴ demonstrates that strategic network pruning can generate capacity for new tasks within fixed architectures.

Replay-based approaches maintain episodic memory buffers containing representative samples from previously learned tasks, interleaving them with current task data during training^{20,16}. Gradient Episodic Memory¹⁶ utilizes constrained optimization to guarantee that parameter updates do not degrade performance on stored previous task examples. The computationally streamlined A-GEM variant¹⁷ relaxes these constraints by requiring only that averaged gradients over memory samples maintain favorable directions, reducing computational complexity from quadratic to linear scaling. The incremental Classifier and Representation Learning framework⁸ integrates memory replay with knowledge distillation techniques, employing nearest-mean-of-exemplars classification during inference. Generative replay methods address privacy considerations through trained generative models that synthesize previous task data. Deep Generative Replay¹⁸ implements a cooperative dual-network architecture of generator and classifier components, where generators produce synthetic replay samples while classifiers perform recognition tasks.

Prototype-driven approaches represent categories through learned vector embeddings, commonly derived as centroids²⁵. These methods provide computational efficiency alongside natural compatibility with metric learning frameworks. Predictions for a target class can be made based on the similarity between these prototypes and new samples' representations. The effectiveness of these methods can be enhanced by accurately maintaining good estimates of previous class prototypes. The overlap between class prototypes is an indication of catastrophic forgetting. To address this class confusion, numerous studies²⁸ utilize self-distillation to retain prior knowledge. However, knowledge distillation requires that adequate discriminative features have been acquired for former classes, and these features continue to be discriminative when learning a new class. To this end, online prototype learning method²⁵ introduces adaptive prototype updating mechanisms that can maintain class representations while accommodating emerging categories.

For explicitly modeling and controlling representational evolution across learning episodes, recent studies introduce energy-based models that learn probability distributions through energy functions that map input configurations to scalar values²⁶. Lower energy values correspond to higher probability densities, enabling flexible modeling of complex data distributions. They illustrate how energy landscapes can characterize feature distributions from distinct tasks and identify representational drift that would otherwise compromise performance on previous tasks.

2.2. Class Incremental Learning (CIL)

As alluded before, CIL exposes another great challenge, i.e. the inter-task class separation (ICS) occurring as the learner lacks access to the training dataset of former tasks when learning a new task. Recent studies underpin three conditions necessary and sufficient for a robust and effective CIL model. They are (i) proficient within-task prediction (WP), (ii) proficient task-id prediction (TP) and (iii) excellent out-of-distribution (OOD) detection for each task.

For a training set $D = \{(x^i, y^i)_{i=1}^n\}$ of each task, where n is the number of data samples, $x^i \in \chi$ is an input sample and $y^i \in \gamma$ is its corresponding label. γ is the set of all labels in D and is referred to as the in-distribution (IND). The goal of OOD detection is to construct a function $f : \chi \rightarrow \gamma \cup \{O\}$ that can identify Out-of-Distribution (OOD) instances not belonging to any class in γ . The class of such instances is denoted as O . When the model can effectively recognize OOD instances for each task, it hence addresses the inter-task class separation (ICS) problem²⁹. ICS refers to the situation in which the learner encounters challenges in establishing clear boundaries between classes of different tasks.

Regarding WP and TP, Kim et al.^{29,30}, present a formula for calculating the probability of a test sample belonging to a task, which is the product of two probabilities, i.e. WP and TP as in formula 1 and 2 as below, respectively.

$$\mathbf{P}_{WP} = \mathbf{P}(x \in X_{k,j} | x \in \mathbf{X}_k, D), \quad (1)$$

$$\mathbf{P}_{TP} = \mathbf{P}(x \in \mathbf{X}_k | D), \quad (2)$$

where $\mathbf{X}_k$ is the dataset of task k , $\mathbf{Y}_k$ is the set of classes for task k and $\mathbf{X}_k = \{\mathbf{X}_{k,j}\}$, j represents the j^{th} class of the k^{th} task. The classes within each task are distinct, i.e., $\mathbf{X}_{k,j} \cap \mathbf{X}_{k,j'} = \emptyset, \forall j \neq j'$. The tasks are also distinct, i.e., $\mathbf{X}_k \neq \mathbf{X}_{k'} = \emptyset, \forall k \neq k'$. In this line, the prediction probability of a CIL model is given by formula 3 as follows:

$$\begin{aligned} \mathbf{P}(x \in \mathbf{X}_{k_0,j_0} | D) &= \sum_{k=1, \dots, n} \mathbf{P}(x \in X_{k,j_0} | x \in \mathbf{X}_k, D) \mathbf{P}(x \in \mathbf{X}_k | D) \\ &= \mathbf{P}(x \in X_{k_0,j_0} | x \in \mathbf{X}_{k_0}, D) \mathbf{P}(x \in \mathbf{X}_{k_0} | D), \end{aligned} \quad (3)$$

Kim et al., 2023 proposed a continual learning method implemented with a replay-based Class Incremental Learning called ROW³⁰ (ROW is an abbreviation for Replay, OOD, WP). At k^{th} task, the system receives the dataset D_k and leverages the replayed data stored from previous tasks in the memory buffer M as out-of-distribution data for task k to train an OOD detection head while adjusting a WP head simultaneously. Specifically, the model for each task consists of two heads: one head for OOD to compute TP and another head for WP. The model trains and fine-tunes the parameters ψ_k of the feature extraction function f_k , the parameters θ_k of the OOD head h_k , and the parameters ϕ_k of the WP head similar to a classification head g_k . Both h_k and g_k heads receive the same features extracted by f_k to calculate the corresponding probabilities. Recently, energy-based models (EBMs)³¹ have been used for OOD detection^{32,33}.

3. Methodology

3.1. Problem Definition

Continual learning (CL) handles a single-pass data stream $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_T\}$ to learn sequentially T tasks. The t^{th} task has a training dataset $\mathcal{D}_t$ consisting of N_t samples and a non-overlapping class label set $\mathcal{C}_t$, i.e., $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t}$, $y_i \in \mathcal{C}_t$ is the label of x_i .

The prominent replay-based approach for CL utilizes a memory bank $\mathcal{M}$ to keep some samples from previously learned tasks. At each time step while training task t , the model learns from a mini-batch data of two components: X uniformly drawn from $\mathcal{D}_t$, and X^b uniformly drawn from $\mathcal{M}$. During the inference phase within the class-incremental continual learning, the model does not know which task the data comes from. The goal of CL is to train a model on the single-pass data stream $\mathcal{D}$ that performs well on both current and previous classes (or tasks).

3.2. Overall model architecture

The unified workflow diagram of our proposed method is illustrated in Fig. 1. The workflow consists of four main phases for processing input data through a probabilistic model. In **Phase 1**, the input data (X, X^b) is first processed through an encoder f_i (optimized with loss $\mathcal{L}_{APF}$), followed by a projection function g (optimized with loss $\mathcal{L}_{onpro}$) to generate the embedding z . The embedding z then flows into two parallel processing streams. **Phase 2** ($\mathcal{L}_{WP}$) consists of multiple within-prototype components ($wp_1, wp_i, \dots, wp_T$) that process the embedding in parallel. Subsequently, **Phase 3** ($\mathcal{L}_{ood}$) applies out-of-distribution detection through multiple OOD detectors ($ood_1, ood_i, \dots, ood_T$), where each detector processes the output from its corresponding within-prototype component. The outputs from all OOD detectors are then combined to compute the final probability distribution $P(X_{k,j}|x) = P(X_{k,j}|z_{e^*}, k) \cdot P(X_k|z_a)$. Simultaneously, **Phase 4** (Training ELI - Energy Alignment) operates on the right side of the workflow. This phase takes the current embedding z_t and the previous embedding z_{t-1} as inputs. The current embedding z_t produces a high-energy representation e_t , while the previous embedding z_{t-1} generates a lower-energy representation e_{t-1} . These energy representations are fed into the ELI model, which performs energy alignment to produce the final aligned embedding e_t , ensuring temporal consistency and energy optimization across sequential embeddings.

Particularly, given a test sample x , the model maps it to an embedding z using the encoder network f and the projector g as follows:

$$\mathbf{z}_x = g(f(\text{aug}(x); \theta_f); \theta_g), \quad (4)$$

and the prediction probability for a class j within each task k is the product of $WP = \mathbf{P}(X_{k,j}|\mathbf{z}_x, k)$ and $TP = \mathbf{P}(X_k|\mathbf{z}_x)$:

$$\mathbf{P}(X_{k,j}|x) = \mathbf{P}(X_{k,j}|\mathbf{z}_x, k)\mathbf{P}(X_k|\mathbf{z}_x), \quad (5)$$

where $X_{k,j}$ denotes class j within task k .

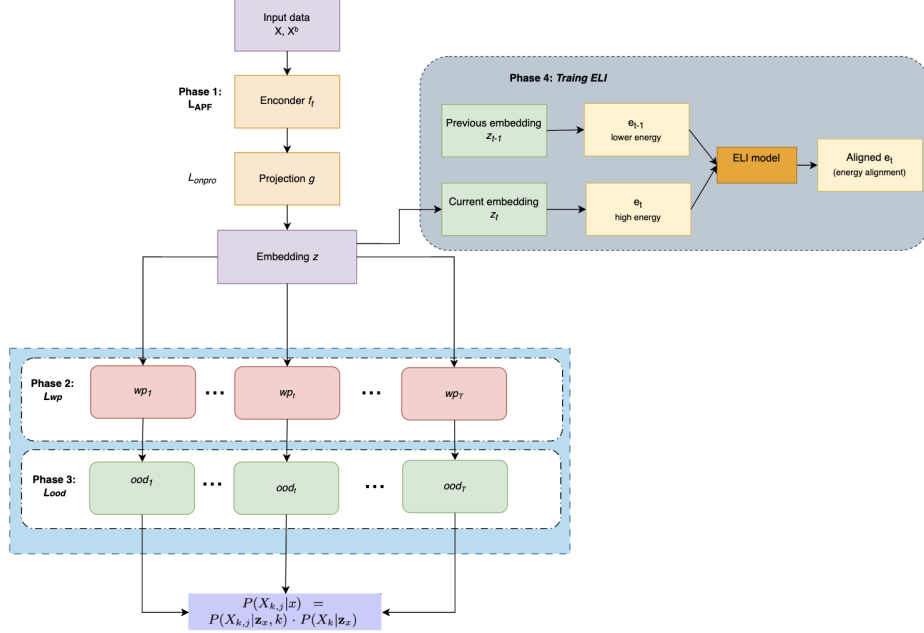


Fig. 1: Our proposed workflow diagram. The workflow encodes input data through phases of embedding generation, within-prototype classification, OOD detection, and energy-based alignment between current and previous embeddings.

The online prototype of each class at each step when learning task t is defined as the mean representation for data in a mini-batch:

$$\mathbf{p}_i = \frac{1}{n_i} \sum_j \mathbf{z}_j \cdot \mathbb{1}\{y_j = i\}, \quad (6)$$

where n_i is the number of samples of class i in the mini-batch, and $\mathbb{1}$ is the indicator function. A set of K online prototypes in X is denoted by $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^K$, and a set of K^b online prototypes in X^b is denoted by $\mathcal{P}^b = \{\mathbf{p}_i^b\}_{i=1}^{K^b}$, where $K = |\mathcal{P}| \leq |\mathcal{C}_t|$, $K^b = |\mathcal{P}^b| \leq \sum_{i=1}^t |\mathcal{C}_i|$, $|\cdot|$ indicates the set cardinal.

To this end, our model comprises four phases, namely (i) Equilibrium training with prototypical feedback, (ii) Fine-tuning all WP-heads, (iii) Fine-tuning all OOD-heads and (iv) Training an ELI model. First, we train the encoder and WP-heads using equilibrium with prototypical feedback. When learning a new task, we add a pair of WP-head and OOD-head to the model. Then, we freeze the encoder and fine-tune all WP-heads using the samples from the memory bank $\mathcal{M}$. Next, we fine-tune all OOD-heads. Finally, we train the ELI model. The training process is illustrated in figure 7. Our model is described in algorithm 3.1.

Algorithm 3.1 Our training algorithm**Require:** Dataset: $\mathcal{D}$; Encoder network: f ; Projection head: g **Ensure:** Trained model

```

1: Initialize  $f$  with parameters trained on ImageNet
2: for  $t \leftarrow 1, T$  do
3:   Add  $wp_t$   $ood_t$ 
4:   for  $e \leftarrow 1, epoch_1$  do
5:     for mini-batch  $X$  in  $\mathcal{D}_t$  do
6:        $X^b \leftarrow \text{APF}(\mathcal{M})$ 
7:        $\hat{X}, \hat{X}^b \leftarrow \text{aug}(X, X^b)$ 
8:        $\mathbf{z}, \mathbf{z}^b \leftarrow g(f(X \cup \hat{X})), g(f(X^b \cup \hat{X}^b))$ 
9:       Calculate online prototypes  $\mathcal{P}$  and  $\mathcal{P}^b$  ▷ Refer equation 4
10:       $\mathcal{L}_{\text{OnPro}} = \mathcal{L}_{\text{OPE}}(\mathcal{P}, \mathcal{P}^b) + \mathcal{L}_{\text{INS}}(\mathbf{z}, \mathbf{z}^b) + \mathcal{L}_{\text{CE}}(wp_t(f(\hat{X}^b)))$ 
11:       $\theta_f, \theta_g, \theta_{wp} \leftarrow \mathcal{L}_{\text{OnPro}}$ 
12:       $\mathcal{M} \leftarrow \text{Update}(\mathcal{M}, X)$ 
13:     end for
14:   end for
15:   if  $t > 1$  then
16:     for  $i \leftarrow 1, t$  do
17:        $D \leftarrow \text{Take from } \mathcal{M} \text{ samples of task } i$ 
18:       for  $e \leftarrow 1, epoch_2$  do
19:         for mini-batch  $X$  in  $D$  do
20:            $\mathcal{L}_{\text{WP}} = \text{CE}(wp_i(f(X)))$ 
21:            $\theta_{wp_i} \leftarrow \mathcal{L}_{\text{WP}}$ 
22:         end for
23:       end for
24:     end for
25:     for  $i \leftarrow 1, t$  do
26:       for  $e \leftarrow 1, epoch_3$  do
27:         for mini-batch  $X$  in  $\mathcal{D}_t$  do
28:            $X^b \leftarrow \text{Sample from } \mathcal{M}$ 
29:            $\mathcal{L}_{\text{TP}} = \text{CE}(ood_i(f(X \cup X^b)))$ 
30:            $\theta_{ood_i} \leftarrow \mathcal{L}_{\text{TP}}$ 
31:         end for
32:       end for
33:     end for
34:     Training ELI model follows the algorithm 3.2
35:   end if
36:    $\mathcal{M} \leftarrow \text{Update}(\mathcal{M}, X)$ 
37: end for

```

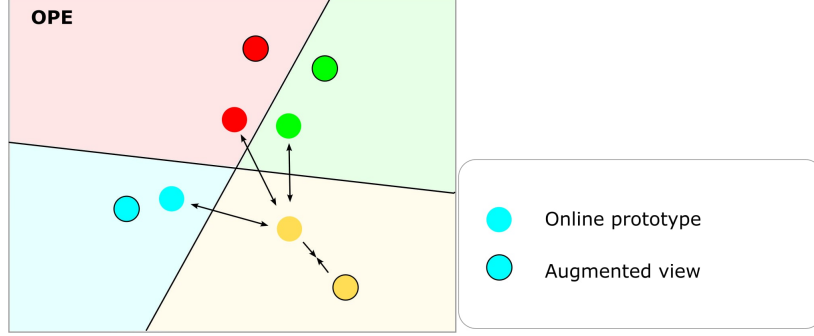


Fig. 2: Visualization of Online prototype equilibrium (OPE), in which representations of points within the same class are pushed close together, while those from different classes are pushed far apart (in the direction of the arrow), as indicated by the arrows' directions.

3.3. Equilibrium Training with Prototypical Feedback

To enhance the model's capacity for learning discriminative class features in a continual learning context, we employ the method Online Prototype Equilibrium (OPE) from ²⁵. This method aims to simultaneously cultivate multiple closer representations for each class while maintaining clear distinctions between classes, as illustrated in Figure 2. The OPE process is formalized through a contrastive loss function:

$$\ell(\mathcal{P}, \hat{\mathcal{P}}) = \frac{-1}{|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} \log \frac{\exp\left(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_i}{\tau}\right)}{\sum_j \exp\left(\frac{\mathbf{p}_i^T \hat{\mathbf{p}}_j}{\tau}\right) + \sum_{j \neq i} \exp\left(\frac{\mathbf{p}_i^T \mathbf{p}_j}{\tau}\right)}, \quad (7)$$

In this equation, τ serves as a temperature parameter, while $\mathcal{P}$ and $\hat{\mathcal{P}}$ represent ℓ_2 -normalized sets of class representations (or online prototypes, according to ²⁵) and their augmented counterparts. To ensure a comprehensive contrastive loss calculation, the final loss $\mathcal{L}_{\text{pro}}$ over online prototypes is defined as:

$$\mathcal{L}_{\text{pro}}(\mathcal{P}, \hat{\mathcal{P}}) = \frac{1}{2} \left[\ell(\mathcal{P}, \hat{\mathcal{P}}) + \ell(\hat{\mathcal{P}}, \mathcal{P}) \right]. \quad (8)$$

To address the dual challenges of acquiring knowledge from new classes and preserving information from previously seen ones, the OPE objective $\mathcal{L}_{\text{pro}}$ is formulated as follows:

$$\mathcal{L}_{\text{OPE}} = \mathcal{L}_{\text{pro}}^{\text{new}}(\mathcal{P}, \hat{\mathcal{P}}) + \mathcal{L}_{\text{pro}}^{\text{seen}}(\mathcal{P}^b, \hat{\mathcal{P}}^b), \quad (9)$$

where $\mathcal{L}_{\text{pro}}^{\text{new}}$ focuses on assimilating information from *new* classes, and $\mathcal{L}_{\text{pro}}^{\text{seen}}$ is dedicated to maintaining the learned knowledge of previously *seen* classes.

OPE tends to treat each class *equally*, not focusing on easily confused classes, which are more likely to be misclassified. We utilized an Adaptive Prototypical Feedback (APF)

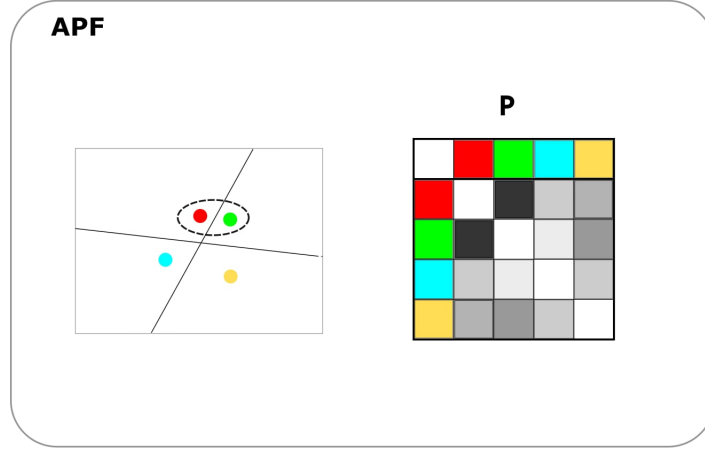


Fig. 3: Illustration of Adaptive Prototypical Feedback (APF), in which the distances between class prototypes are calculated to identify classes that are close to each other, thus easily confused with one another. These distances are used to calculate the replay probabilities for prototypes in the memory bank $\mathcal{M}$. Prototypes of easily confused classes having higher probabilities of being replayed, helping reduce distinction confusion.

strategy²⁵ to guide OPE to overcome this limitation. APF assesses the potential for class confusion by calculating the distances between online prototypes of all pairs of previously encountered classes in the memory bank $\mathcal{M}$, as illustrated in Figure 3. To this end, classes with closer prototypes indicate a higher likelihood of misclassification, thus acquire more replays to enhance their decision boundaries. The prototype distance matrix is then converted into a probability distribution P over all classes. This conversion utilizes a symmetric Gaussian kernel, which can be formally defined as:

$$P_{i,j} \propto \exp(-\|\mathbf{p}_i^b - \mathbf{p}_j^b\|_2^2), \quad (10)$$

for $i, j \in \{1, \dots, |\mathcal{P}^b|\}$ and $i \neq j$. Before being used for guiding the memory sampling process afterward to create a replayed mixup³⁴, all probabilities in P are normalized to a probability mass function that sums to one. To construct replay data X^b of size m , APF first selects $n_{\text{APF}} = \alpha \cdot m$ samples for replay according to the probability distribution P , where a higher $P_{a,b}$, the more likely selected classes a and b . The parameter α determines the proportion of confusion-guided sampling. Then, the replay data is filled up by the remaining $m - n_{\text{APF}}$ samples selected uniformly from the memory bank. These remaining samples help the model not only focus on easily misclassified classes and thus not disrupt the equilibrium established by OPE. As a result, the integration of OPE and APF equips the model with the capability to preserve the discrimination across all previously encountered classes as learning representative features for new classes.

While class-level online prototypes offer a high-level representation, they may not fully

capture the full spectrum of variations within each class, potentially leading to less compact class representations. To address this limitation and learn more comprehensive intra-class representations, supervised contrastive learning is utilized through the following formulation:

$$\begin{aligned} \mathcal{L}_{\text{INS}} = & \sum_{i=1}^{2N} \frac{-1}{|I_i|} \sum_{j \in I_i} \log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)/\tau')}{\sum_{k \neq i} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau')} \\ & + \sum_{i=1}^{2m} \frac{-1}{|I_i^b|} \sum_{j \in I_i^b} \log \frac{\exp(\text{sim}(\mathbf{z}_i^b, \mathbf{z}_j^b)/\tau')}{\sum_{k \neq i} \exp(\text{sim}(\mathbf{z}_i^b, \mathbf{z}_k^b)/\tau')}, \end{aligned} \quad (11)$$

where $I_i = \{j \in \{1, \dots, 2N\} \mid j \neq i, y_j = y_i\}$ and $I_i^b = \{j \in \{1, \dots, 2m\} \mid j \neq i, y_j^b = y_i^b\}$ indicate the sets of positive sample indices for $\mathbf{z}_i$ and $\mathbf{z}_i^b$, respectively. y_i^b denotes the class label of input x_i^b from X^b . N indicates the batch size of the current input X . The temperature hyper-parameter is denoted by τ' . The cosine similarity is employed for the sim.

To this end, the total loss of the equilibrium training with prototypical feedback is as follows²⁵:

$$\mathcal{L}_{\text{OnPro}} = \mathcal{L}_{\text{OPE}} + \mathcal{L}_{\text{INS}} + \mathcal{L}_{\text{CE}}, \quad (12)$$

where $\mathcal{L}_{\text{CE}} = \text{CE}(y^b, \varphi(f(\text{aug}(x^b))))$ denotes the conventional cross-entropy loss. This multi-faceted approach synergistically combines class-level prototype learning, instance-level similarity optimization, and traditional classification objectives to effectively navigate the complexities of continual learning scenarios.

3.4. Multi-Head Structure

3.4.1. Fine-tuning WP-heads of all tasks

As the encoder network is trained for the current task, the features extracted from it might not be accurate when passed through WP-heads of old tasks. Thus, we have to fine-tune WP heads for all learned tasks. We take samples of each task from the memory bank $\mathcal{M}$. After training for the encoder network, we only freeze its parameters and train for WP-heads, as illustrated in Figure 4. We only take samples from old tasks in $\mathcal{M}$ to train by minimizing cross-entropy loss:

$$\mathcal{L}_{\text{WP}} = -\frac{1}{N} \sum_{(x,y) \in \mathcal{M}} \log p(y|x, t), \quad (13)$$

3.4.2. Fine-tuning OOD-heads of all tasks

Similar to WP-heads, OOD-heads are biased. We fine-tune them by freezing the encoder network and training for only OOD-heads, as illustrated in Figure 5. We use samples from

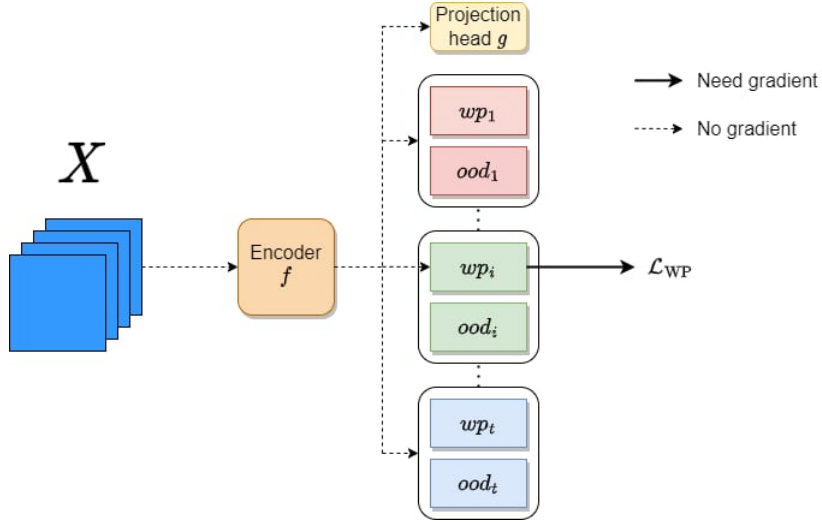


Fig. 4: Fine-tune WP-heads

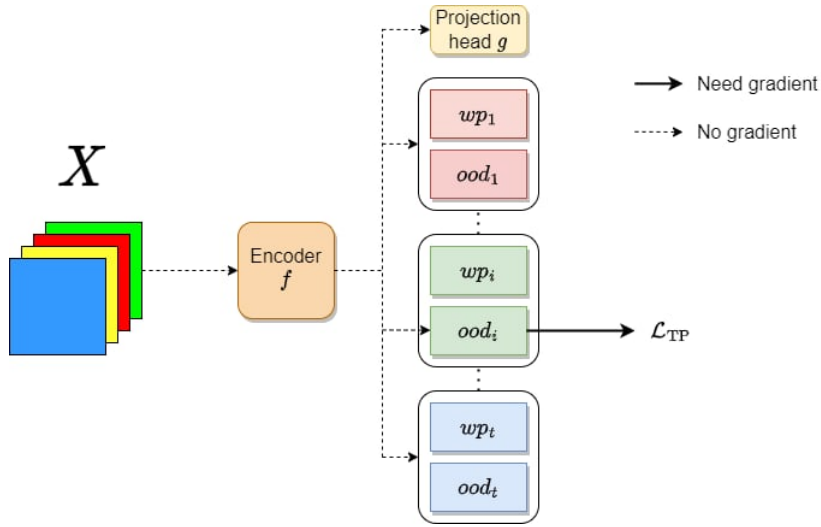


Fig. 5: Fine-tune OOD-heads

both current task $\mathcal{D}_t$ and memory bank $\mathcal{M}$ to train by minimizing cross-entropy loss:

$$\mathcal{L}_{TP} = -\frac{1}{M} \left(\sum_{(x,y) \in \mathcal{D}_t} \log p(y|x, t) + \sum_{(x,y) \in \mathcal{M}} \log p(y|x, t) \right), \quad (14)$$

where $M = |\mathcal{D}_t| + |\mathcal{M}|$.

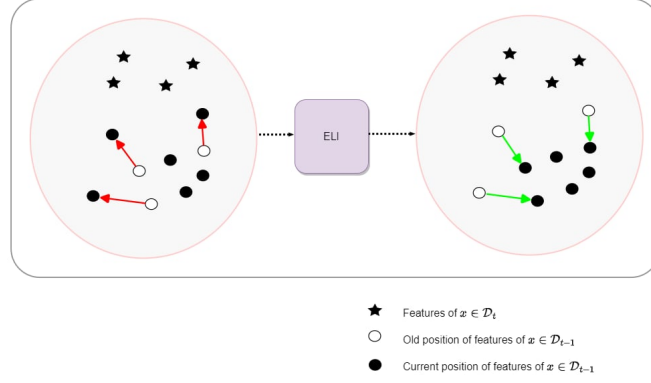


Fig. 6: Energy-based latent aligner. The feature representations are optimized for the circle recognition task when trained on the circular dataset. However, when learning an additional task of square recognition, the hidden feature space of circles shifts closer to stars (as indicated by the red arrow, shifting in the wrong direction). Then, the ELI model is used to align and pull them back towards each other along the direction of the green arrow.

3.5. Energy-Based Latent Aligner

Energy-based latent aligner (ELI) is a novel methodology that helps counter the representational shift in the latent space of incremental learning models. ELI learns an energy manifold for the latent representations such that previous task latents have low energy and the current task latents have high energy values (see Figure 6).

Once all WP-heads and OOD-heads are fine-tuned, we train the ELI models. We use three components: samples of current task $\mathcal{D}_t$, features of x encoded by the encoder network trained until previous task $\mathbf{e}_{t-1} = f_{t-1}(x)$, and features of x encoded by the encoder network trained until current task $\mathbf{e}_t = f_t(x)$. ELI learns to assign a small energy value for $\mathbf{e}_{t-1}$ and a high energy value for $\mathbf{e}_t$.

We train a model $E_\psi(\mathbf{e}) : \mathbb{R}^D \rightarrow \mathbb{R}$ to map the feature $\mathbf{e} \in \mathbb{R}^D$ to energy value. An energy-based model (EBM) is defined as a Gibbs distribution $p_\psi(\mathbf{e})$ trên $E_\psi(\mathbf{e})$:

$$p_\psi(\mathbf{e}) = \frac{\exp(-E_\psi(\mathbf{e}))}{\int_{\mathbf{e}} \exp(-E_\psi(\mathbf{e})) d\mathbf{e}}, \quad (15)$$

EBM learns to maximize the log-likelihood of a sample taken from the distribution $p_{true}(\mathbf{e})$:

$$L(\psi) = \mathbb{E}_{\mathbf{e} \sim p_{true}} [\log p_\psi(\mathbf{e})]. \quad (16)$$

The derivative of $L(\psi)$ is:

$$\partial_\psi L(\psi) = \mathbb{E}_{\mathbf{e} \sim p_{true}} [-\partial_\psi E_\psi(\mathbf{e})] + \mathbb{E}_{\mathbf{e} \sim p_\psi} [\partial_\psi E_\psi(\mathbf{e})]. \quad (17)$$

The first term in 17 assures that the energy for $\mathbf{e}$ taken from the distribution p_{true} is minimized. The second one maximizes the samples from the current model. In ELI, p_{true}

is the distribution of features extracted from the model trained until the previous task. Taking samples from $p_\psi(\mathbf{x})$ is difficult because of the normalization constant in equation 15. Approximate samples are recursively drawn using Langevin dynamics^{35,36}, which is a popular MCMC algorithm,

$$\mathbf{e}_{i+1} = \mathbf{e}_i - \frac{\lambda}{2} \partial_{\mathbf{e}} E_\psi(\mathbf{e}) + \sqrt{\lambda} \omega_i, \omega_i \sim \mathcal{N}(0, \mathbf{I}) \quad (18)$$

This equation ensures the Markov chain stabilizes to a stationary distribution within a few iterations, starting from an initial value $\mathbf{e}_i$.

Algorithm 3.2 Training ELI

Require: Encoder network trained until current task: f_t ; Encoder network trained until previous task: f_{t-1} ; Data of the current task: $\mathcal{D}_t$

Ensure: E_ψ

- 1: Initialize E_ψ
 - 2: **while** until required iterations **do**
 - 3: $\mathbf{x} \sim \mathcal{D}_t$ ▷ Sample a mini-batch
 - 4: $\mathbf{e}_{t-1} \leftarrow f_{t-1}(\mathbf{x})$
 - 5: $\mathbf{e}_t \leftarrow f_t(\mathbf{x})$
 - 6: $\mathbf{e}_t \leftarrow$ Sample from EBM with $\mathbf{e}_t$ ▷ Refer equation (18)
 - 7: $\text{in_dist_energy} \leftarrow E_\psi(\mathbf{e}_{t-1})$
 - 8: $\text{out_dist_energy} \leftarrow E_\psi(\mathbf{e}_t)$
 - 9: $\text{Loss} \leftarrow (-\text{in_dist_energy} + \text{out_dist_energy})$ ▷ Refer equation (17)
 - 10: Optimize E_ψ with Loss.
 - 11: **end while**
-

After learning task t , we use 3.2 to learn the energy manifold. Then, we use this manifold to align the latent representations of previous task instances from the current model using 3.3.

Algorithm 3.3 Align latents using ELI trained model

Require: Features: $\mathbf{e}$; EBM: E_ψ ; Number of Langevin iterations: L_{steps} ; Learning rate: λ

Ensure: $\mathbf{e}$

- 1: Initialize E_ψ
 - 2: **while** until L_{steps} iterations **do**
 - 3: $\text{grad} = \nabla_{\mathbf{e}} E_\psi(\mathbf{e})$
 - 4: $\mathbf{e} \leftarrow \mathbf{e} - \lambda * \text{grad}$
 - 5: **end while**
-

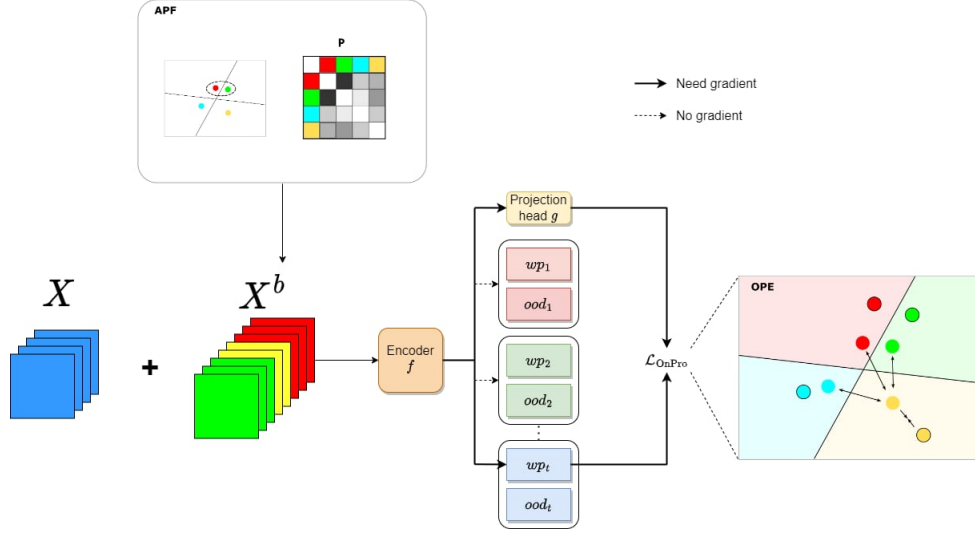


Fig. 7: Overall our proposed model architecture at training phase

4. Experiments and Results

4.1. Experiment Settings

We evaluate our approach across multiple datasets: CCH5000 for medical image classification³⁷, and CIFAR-10/100 for general computer vision tasks. The CCH5000 dataset comprises 5000 colorectal cancer histology images spanning eight tissue types (see Figure 8 for some examples). We employ an 80-20 train-test split with random partitioning, training initially on four classes before incremental learning on remaining classes across two or four tasks, depending on the experimental configuration.

For our proposed method, we adopt hyperparameter settings following established continual learning protocols and preliminary validation experiments, including^{25,30}. Specifically: EBM latent dimension is set to 64, EBM architecture uses 2 layers with 64 hidden units, EBM training runs for 1500 iterations with learning rate 1×10^{-4} , Langevin dynamics sampling employs 30 iterations with learning rate 0.1, Exponential Moving Average (EMA) decay factor is 0.89, the ratio of APF α is set to 0.25 or 0.1 for CIFAR-10, or CIFAR-100, respectively, temperature parameters τ and τ' is set to 0.5 and 0.07 for contrastive learning, and fine-tuning procedures use 5 WP heads and 10 OOD heads per task.

To ensure fair comparison, our implementation employs identical ResNet-18 backbone initialization (ImageNet pre-trained weights, optimized via Adam with learning rate 5×10^{-4}), identical memory buffer constraints, and task-specific hyperparameters as reported in their original publications. When baseline implementations require dataset-specific tuning (e.g., learning rates, batch sizes), we adopt the values recommended by the respective authors or replicate their reported experimental configurations. Particularly, training batch configurations include current task samples ($N = 10$) and replay buffer

samples ($m = 32$), maintaining consistent batch composition across all experiments.

All comparisons maintain identical backbone initialization and memory constraints. Results are averaged across three independent runs with different random seeds to account for stochastic variability. Performance assessment employs two complementary metrics, namely average accuracy and average forgetting.

Average Accuracy (ACC) evaluates the average accuracy of the model across all tasks:

$$ACC = \frac{1}{T} \sum_{j=1}^T a_{T,j}, \quad (19)$$

where $a_{T,j}$ is the accuracy of task j after the model has been trained on all tasks.

Average Forgetting (AF) represents the extent to which the model forgets each task after being trained on the final task and is calculated using the formula:

$$AF = \frac{1}{T-1} \sum_{j=1}^{T-1} \max_{k \leq T-1} (a_{k,j} - a_{T,j}) \quad (20)$$

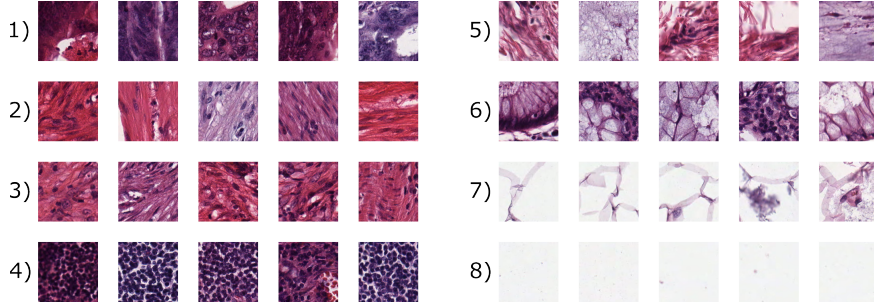


Fig. 8: Here are 5 random images from each class of the CCH5000 dataset, including: 1) tumour epithelium, 2) simple stroma, 3) complex stroma (stroma that contains single tumour cells and/or single immune cells), 4) immune cell conglomerates, 5) debris and mucus, 6) mucosal glands, 7) adipose tissue, 8) background

4.2. Results

4.3. Medical Image Classification Results

We compare our model's results to several state-of-the-art methods for solving incremental learning problems. The experiments are implemented in two settings: one and two classes per task. We conducted each experiment three times and averaged the results.

Performance Analysis: Table 1 demonstrates that our model achieves significant improvements in both accuracy and forgetting metrics. With two classes per task, our method achieves the best forgetting score of 2.9, representing a 1.1 point (27.5%) improvement

Table 1: Comparative results on CCH5000 dataset with one and two classes per task settings.

Classes per task	1		2	
Metrics	Acc	Fgt	Acc	Fgt
iCaRL ⁸	91.1	9.0	93.0	6.8
UCIR ³⁸	92.0	5.5	93.9	4.4
PODNet ³⁹	89.2	6.0	92.0	5.2
DER ⁴⁰	91.0	5.6	93.0	6.4
LO2LN ⁴¹	94.5	3.9	94.6	4.0
Onpro ²⁵	89.5	3.6	92.8	3.8
Ours	91.6	4.6	94.6	2.9

over LO2LN⁴¹ and a substantial 3.9 (57.4%) point improvement over iCaRL⁸. The average accuracy matches the best performer (LO2LN) at 94.6% and exceeds PODNet³⁹ by 2.6%.

However, our method shows reduced effectiveness in the single class per task scenario, achieving 91.6% accuracy compared to LO2LN’s 94.5%. This performance degradation is primarily attributed to the limited discriminative power of ELI when each task contains insufficient class diversity for effective energy manifold learning.

Computational Overhead Analysis: Our approach introduces additional computational costs through the multi-head architecture and ELI training. Specifically, the training time increases by approximately 82% compared to the baseline OnPro method (34.82 minutes vs. 19.13 minutes for one class per task), while GPU memory consumption remains comparable at 13.4 GB, indicating efficient parameter sharing in our architecture.. The ELI component alone contributes approximately 8% of the total training time through its Langevin dynamics sampling process.

4.3.1. Benchmark Dataset Evaluation

Our model achieves the highest effectiveness under memory constraints, compared to several other baseline methods in terms of average accuracy, as demonstrated in Table 2. When using the multi-head architecture, our model outperforms Onpro that uses a single head, with experiments on the CIFAR-10 showing an increase of nearly 9% from 52.4% to 63.62% when $\mathcal{M}=100$, and a 6% increase when $\mathcal{M}=200$, and 5% when $\mathcal{M}=500$. The experimental results on CIFAR-100 also show significant improvements, increasing by approximately 7%, 3%, and 1.3% respectively for $\mathcal{M}=500$, $\mathcal{M}=1000$, and $\mathcal{M}=2000$. This pattern reveals our architecture’s particular strength in resource-limited scenarios where replay diversity is insufficient. Further, performance gains correlate inversely with buffer size, suggesting our multi-head architecture effectively compensates for limited replay samples.

The results in Table 3 indicate that, despite achieving superior average accuracy, our proposed model occasionally faces exhibits increased forgetting. For example, in exper-

Table 2: Average accuracy of our model and SOTA models with different buffer sizes $\mathcal{M}$

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL ⁸	31.0	33.9	42.0s	12.8	16.5	17.6
DER++ ¹⁹	31.5	39.7	50.9	16	21.4	23.9
PASS ⁴²	33.7	33.7	33.7	7.5	7.5	7.5
AGEM ¹⁷	17.7	17.5	17.5	5.8	5.9	5.8
GSS ⁴³	18.4	19.4	25.2	8.1	9.4	10.1
ER ⁴⁴	19.4	20.9	26.0	8.7	9.9	10.7
MIR ⁴⁵	20.7	23.5	29.9	9.7	11.2	13.0
GDumb ⁴⁶	23.3	27.1	34	8.2	11	15.3
ASER ⁴⁷	20.0	22.8	31.6	11.0	13.5	17.6
SCR ⁴⁸	40.2	48.5	59.1	19.3	26.5	32.7
CoPE ²	33.5	37.3	42.9	11.6	14.6	16.8
DVC ⁴⁹	35.2	41.6	53.8	15.4	20.3	25.2
OCM ²⁸	47.5	59.6	70.1	19.7	27.5	34.4
OnPro ²⁵	52.4	63.0	67.7	20.7	27.2	34.9
Ours	63.6	69.3	72.8	27.1	30.3	36.3

iments on CIFAR-10, with $\mathcal{M}=100$ and $\mathcal{M}=500$, our model has the smallest forgetting scores, but with $\mathcal{M}=200$, it is larger compared to Onpro (a smaller forgetting score is better). For CIFAR-100, our model only forgets the least with $\mathcal{M}=1000$, while in the other two cases, it is higher than Onpro. This is explained by the fact that our method is currently focusing on optimizing for classification ability of the model and has not addressed the issue of catastrophic forgetting as much.

We monitor task accuracy during the training process, illustrating that our method consistently outperforms Onpro with higher results (see Figure 9).

4.4. Ablation Study

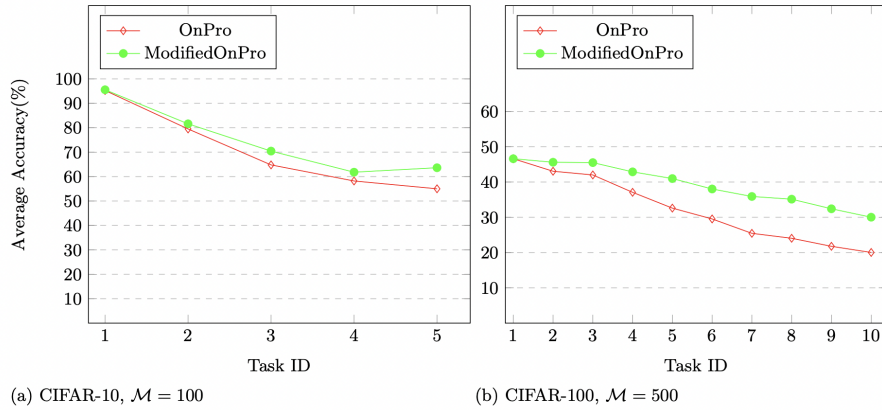
We systematically decompose our method’s components (i.e. WP-heads/OOD-heads and ELI) to identify individual contributions and their interactions. Table 4 reveals distinct component roles:

ELI Impact: ELI removal causes 1.3% accuracy drop and substantial forgetting increase (from 2.9 to 8.5, a 193% increase) in two-class scenarios, demonstrating ELI’s critical role in representational stability. Conversely, single-class configurations show negligible ELI benefit for accuracy (91.6% vs 91.8%) and minimal forgetting impact (4.6 vs 4.8). This limitation is attributed to contrastive loss dynamics: with a buffer containing k classes, the single-class per task setting provides only $k + 1$ classes for contrastive learn-

Table 3: Average Forgetting of our model and SOTA models with different buffer sizes $\mathcal{M}$. Note that lower forgetting indicates better performance.

Method	CIFAR-10			CIFAR-100		
	$\mathcal{M} = 100$	$\mathcal{M} = 200$	$\mathcal{M} = 500$	$\mathcal{M} = 500$	$\mathcal{M} = 1000$	$\mathcal{M} = 2000$
iCaRL ⁸	52.7	49.3	38.3	16.5	11.2	10.4
DER++ ¹⁹	57.8	46.7	33.6	41.0	34.8	33.2
PASS ⁴²	21.2	21.2	21.2	10.6	10.6	10.6
AGEM ¹⁷	64.8	64.8	64.5	41.7	41.8	41.7
GSS ⁴³	67.1	65.8	61.2	48.7	46.7	44.7
ER ⁴⁴	64.7	62.9	57.5	47.0	46.4	44.7
MIR ⁴⁵	62.6	58.5	51.1	45.7	44.2	42.3
GDumb ⁴⁶	28.5	28.4	28.1	25.0	23.2	20.7
ASER ⁴⁷	64.8	62.6	53.2	52.8	50.4	46.8
SCR ⁴⁸	43.2	35.5	24.1	29.3	20.4	11.5
CoPE ²	49.7	45.7	39.4	25.6	17.8	14.4
DVC ⁴⁹	40.2	31.4	21.2	32.0	32.7	28.0
OCM ²⁸	35.5	23.9	13.5	18.3	15.2	10.8
OnPro ²⁵	24.2	16.5	13.7	17.8	14.9	10.0
Ours	23.2	18.3	13.5	19.2	14.5	11.2

Fig. 9: Task accuracy metrics during the training process of our method (green line) vs. the Onpro method (red line).



ing, severely limiting negative sample diversity. The contrastive loss in Algorithm 3.2 (Line 9) requires sufficient negative samples in the denominator for effective gradient computa-

Table 4: Systematic ablation analysis with computational profiling (GPU in GB and Time in minutes).

Classes per task	1				2			
Model	Acc	Fgt	GPU	Time	Acc	Fgt	GPU	Time
Ours	91.6	4.6	13.4	34.8	94.6	2.9	13.4	25.3
– ELI	91.8	4.8	13.4	33.2	93.3	8.5	13.4	24.1
– <i>WP, OOD</i>	89.7	2.3	13.4	20.3	92.1	5.7	13.4	18.8
– ELI (Onpro ²⁵)	89.5	3.6	13.4	19.1	92.8	3.8	13.4	18.1

tion and meaningful energy manifold construction. Insufficient class diversity results in poor energy separation between current and previous task representations, reducing ELI’s discriminative power. When excluding all three components (i.e., WP, OOD, and ELI), the model degenerates to the baseline OnPro model, showing accuracy decreases of 2.1 percentage points (from 91.6 to 89.5) and 1.8 percentage points (from 94.6 to 92.8) in one-class and two-class settings, respectively, confirming the cumulative contribution of our proposed components.

Multi-head Architecture Contribution: WP/OOD head removal yields consistent 1.9–2.5% accuracy degradation across configurations (from 91.6 to 89.7 in one-class; from 94.6 to 92.1 in two-class), demonstrating the architectural importance for classification performance. The multi-head design provides task-specific decision boundaries through specialized feature spaces: WP heads learn within-task class discrimination, while OOD heads explicitly model task boundaries for inter-task separation.

However, the ablation study reveals a nuanced accuracy-forgetting tradeoff. Removing WP/OOD heads actually reduces forgetting in certain configurations: from 4.6 to 2.3 in the one-class setting and from 2.9 to 5.7 in the two-class setting (though the latter shows increased forgetting). This counterintuitive result in the one-class scenario suggests that the multi-head architecture, while improving accuracy through specialized task representations, may introduce additional plasticity that increases susceptibility to representational drift when combined with limited class diversity. The single-head baseline (OnPro) achieves 3.6 forgetting rate in this setting—intermediate between our full model (4.6) and the model without multi-heads (2.3)—indicating that the interaction between multi-head architecture and ELI is not simply additive.

In the two-class setting, the synergy between components becomes more apparent: the full model achieves the best forgetting metric (2.9), outperforming both the model without ELI (8.5) and without multi-heads (5.7). This suggests that multi-head architecture and energy-based alignment function as complementary mechanisms when sufficient class diversity enables effective contrastive learning. The WP/OOD heads create structured task-specific representations, while ELI stabilizes these representations across learning episodes, preventing catastrophic interference.

Regarding computational efficiency, our analysis reveals asymmetric contributions from the two components. Using only the ELI component (i.e., excluding multi-head ar-

chitecture) achieves substantial training efficiency: 20.3 minutes and 18.8 minutes for one-class and two-class settings, respectively, compared to 34.8 and 25.3 minutes for the full model. This represents absolute overhead of 14.5 minutes (71.4% increase) and 6.5 minutes (34.6% increase) attributable to the multi-head architecture. The overhead stems from the iterative fine-tuning procedure, where each task-specific WP and OOD head must be independently optimized after encoder training.

Conversely, using only the multi-head component (i.e., excluding ELI) requires 33.2 and 24.1 minutes—representing absolute overhead of only 1.6 minutes (4.8%) and 1.2 minutes (5.0%) for ELI. Comparing against the baseline OnPro (19.1 and 18.1 minutes), adding multi-heads increases training time by 14.1 minutes (73.8%) and 6.0 minutes (33.1%), while adding ELI to the multi-head configuration adds only 1.6 minutes (4.8%) and 1.2 minutes (5.0%).

This computational decomposition reveals that multi-head architecture accounts for approximately 93% (14.5/15.7) and 91% (6.5/7.2) of the total overhead in one-class and two-class settings, respectively, while ELI contributes the remaining 7% and 9%. The larger relative ELI cost in two-class scenarios aligns with its performance profile: ELI provides minimal benefit in single-class configurations but delivers substantial forgetting reduction (65.9%, from 8.5 to 2.9) in multi-class settings where contrastive learning is more effective.

From a cost-benefit perspective, the full model’s superior combined performance (94.6% accuracy with 2.9 forgetting in two-class settings) justifies the computational investment when both metrics are critical. However, practitioners facing strict computational constraints might consider using only multi-heads (achieving 93.3% accuracy with 95.4% of full training time) or only ELI (achieving 92.1% accuracy with 74.3% of full training time), depending on whether accuracy or training efficiency is prioritized.

5. Discussion

Memory-Free Adaptation Impossibility: Our approach fundamentally depends on replay buffers for prototypical feedback and energy manifold learning. Adaptation to memory-free settings would require replacing replay-based components with parameter regularization or generative models, fundamentally altering the method’s core mechanisms and likely sacrificing performance advantages.

Computational Scalability Barriers: Linear parameter growth with task number creates practical deployment limitations. Our design requires $O(T)$ parameter growth for T tasks, creating memory scaling challenges. Each task adds approximately $2 \times d_{\text{hidden}}$ parameters for WP/OOD heads, becoming prohibitive beyond 20-25 tasks in standard GPU configurations. Consequently, this necessitates architectural innovations like shared head mechanisms or automated pruning strategies.

Representational Overlap Sensitivity: ELI effectiveness diminishes when task representations overlap significantly. Energy manifold learning requires distinguishable feature distributions; excessive overlap creates unstable energy landscapes, explaining performance degradation in challenging continual learning scenarios.

6. Conclusion

In this paper, we study continual learning in the Class Incremental Learning setting. To this end, we proposed a replay-based method that is built upon the OnPro model²⁵. To tackle the significant challenge of catastrophic forgetting in continual learning, we first propose employing Equilibrium Training with Prototypical Feedback to enhance the quality of hidden features. Secondly, to improve the model's classification performance, for each task, we propose to use: (i) the within-task prediction (WP) head to guide our CF model in recognizing data belonging to that task and (ii) the out-of-distribution (OOD) head to guide the model in identifying data belonging to a specific class within that task. Thirdly, we further improve OnPro by applying an energy-based model from²⁶ to minimize the representational shift in the latent space (in the hidden layers) of the incremental model. We learn an energy manifold where the latent representations from the model trained for the current task have higher energy, while the latent representations from the model trained for the previous task have lower energy. The learned energy-based model helps transform the previous task latents such that the representational shift is undone. Experimental results on benchmark medical and general image classification datasets demonstrate the effectiveness of our proposed method, outperforming existing state-of-the-art models.

References

1. G. I. Parisi, R. Kemker, J. L. Part, C. Kanan and S. Wermter, Continual lifelong learning with neural networks: A review, *Neural Networks* **113** (2019) 54–71.
2. M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh and T. Tuytelaars, A continual learning survey: Defying forgetting in classification tasks, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **44**(7) (2021) 3366–3385.
3. M. McCloskey and N. J. Cohen, Catastrophic interference in connectionist networks: The sequential learning problem, *Psychology of Learning and Motivation* **24** (1989) 109–165.
4. I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville and Y. Bengio, An empirical investigation of catastrophic forgetting in gradient-based neural networks, *arXiv preprint arXiv:1312.6211 N/A* (2013).
5. R. M. French, Catastrophic forgetting in connectionist networks, *Trends in Cognitive Sciences* **3**(4) (1999) 128–135.
6. J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska et al., Overcoming catastrophic forgetting in neural networks, *Proceedings of the National Academy of Sciences* **114**(13) (2017) 3521–3526.
7. G. M. van de Ven, T. Tuytelaars and A. S. Tolias, Three types of incremental learning, *Nature Machine Intelligence* **4**(12) (2022) 1185–1197.
8. S.-A. Rebuffi, A. Kolesnikov, G. Sperl and C. H. Lampert, icarl: Incremental classifier and representation learning, in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, (IEEE, 2017), pp. 2001–2010.
9. Z. Mai, R. Li, J. Jeong, D. Quispe, H. Kim and S. Sanner, Online continual learning in image classification: An empirical survey, *Neurocomputing* **469** (2022) 28–51.
10. L. Wang, X. Zhang, H. Su and J. Zhu, A comprehensive survey of continual learning: Theory, method and application, *IEEE transactions on pattern analysis and machine intelligence* **46**(8) (2024) 5362–5383.
11. F. Zenke, B. Poole and S. Ganguli, Continual learning through synaptic intelligence, in *International Conference on Machine Learning*, (PMLR, 2017), pp. 3987–3995.

12. R. Aljundi *et al.*, A statistical theory of regularization-based continual learning, in *International Conference on Machine Learning*, (PMLR, 2024).
13. A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu and R. Hadsell, Progressive neural networks, in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 523–532.
14. A. Mallya and S. Lazebnik, Packnet: Adding multiple tasks to a single network by iterative pruning, in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, (IEEE, 2018), pp. 7765–7773.
15. R. Aljundi, P. Chakravarty and T. Tuytelaars, Expert gate: Lifelong learning with a network of experts, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (IEEE, 2017), pp. 7120–7129.
16. D. Lopez-Paz and M. Ranzato, Gradient episodic memory for continual learning, in *Advances in Neural Information Processing Systems*, **30** 2017.
17. A. Chaudhry, M. Ranzato, M. Rohrbach and M. Elhoseiny, Efficient lifelong learning with a-gem, in *International Conference on Learning Representations*, 2019.
18. H. Shin, J. K. Lee, J. Kim and J. Kim, Continual learning with deep generative replay, in *Advances in Neural Information Processing Systems*, **30**, (PMLR, 2017), pp. 2990–2999.
19. P. Buzzega, M. Boschini, A. Porrello, D. Abati and S. Calderara, Dark experience for general continual learning: a strong, simple baseline, *Advances in neural information processing systems* **33** (2020) 15920–15930.
20. A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. Dokania, P. Torr and M. Ranzato, Continual learning with tiny episodic memories, in *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019.
21. M. Caccia, P. Rodriguez, O. Ostapenko, F. Normandin, M. Lin, L. Page-Caccia, I. H. Laradji, I. Rish, A. Lacoste, D. Vázquez *et al.*, Online fast adaptation and knowledge accumulation (os-aka): a new approach to continual learning, *Advances in Neural Information Processing Systems* **33** (2020) 16532–16545.
22. N. Asadi, M. Davari, S. Mudur, R. Aljundi and E. Belilovsky, Prototype-sample relation distillation: towards replay-free continual learning, in *International conference on machine learning*, PMLR2023, pp. 1093–1106.
23. R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach and T. Tuytelaars, Memory aware synapses: Learning what (not) to forget, in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 139–154.
24. J. Serra, D. Suris, M. Miron and A. Karatzoglou, Overcoming catastrophic forgetting with hard attention to the task, in *International Conference on Machine Learning*, (PMLR, 2018), pp. 4548–4557.
25. Y. Wei, J. Ye, Z. Huang, J. Zhang and H. Shan, Online prototype learning for online continual learning, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18764–18774.
26. K. Joseph, S. Khan, F. S. Khan, R. M. Anwer and V. N. Balasubramanian, Energy-based latent aligner for incremental learning, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (IEEE, 2022), pp. 7452–7461.
27. Q. T. P. Thi, D. H. Nguyen, T. H. Dang, D. T. Le, T. T. Nguyen and Q. T. Ha, Towards robust continual learning: A multi-head approach with online prototype equilibrium and adaptive prototypical feedback, in *Asian Conference on Intelligent Information and Database Systems*, (Springer Nature Singapore, Singapore, April 2024), pp. 275–286.
28. Y. Guo, B. Liu and D. Zhao, Online continual learning through mutual information maximization, in *International Conference on Machine Learning*, PMLR2022, pp. 8109–8126.
29. G. Kim, C. Xiao, T. Konishi, Z. Ke and B. Liu, A theoretical study on solving continual learning, *Advances in Neural Information Processing Systems* **35** (2022) 5065–5079.

24 Pham et al., 2024

30. G. Kim, C. Xiao, T. Konishi and B. Liu, Learnability and algorithm for continual learning, in *International conference on machine learning*, (PMLR, 2023), pp. 16877–16896.
31. Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato and F. J. Huang, A tutorial on energy-based learning, *Predicting Structured Data* **1**(0) (2006).
32. W. Liu, X. Wang, J. Owens and Y. Li, Energy-based out-of-distribution detection, *Advances in neural information processing systems* **33** (2020) 21464–21475.
33. F. Tonin, A. Pandey, P. Patrinos and J. A. Suykens, Unsupervised energy-based out-of-distribution detection using stiefel-restricted kernel machine, in *2021 International Joint Conference on Neural Networks (IJCNN)*, (IEEE, 2021), pp. 1–8.
34. H. Zhang, M. Cisse, Y. N. Dauphin and D. Lopez-Paz, mixup: Beyond empirical risk minimization, in *International Conference on Learning Representations (ICLR)*, 2018.
35. R. M. Neal et al., Mcmc using hamiltonian dynamics, *Handbook of markov chain monte carlo* **2**(11) (2011) p. 2.
36. M. Welling and Y. W. Teh, Bayesian learning via stochastic gradient langevin dynamics, in *Proceedings of the 28th International Conference on Machine Learning, ICML2011*, pp. 681–688.
37. J. N. Kather, C.-A. Weis, F. Bianconi, S. M. Melchers, L. R. Schad, T. Gaiser, A. Marx and F. G. Zöllner, Multi-class texture analysis in colorectal cancer histology, *Scientific reports* **6**(1) (2016) 1–11.
38. S. Hou, X. Pan, C. C. Loy, Z. Wang and D. Lin, Learning a unified classifier incrementally via rebalancing, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 831–839.
39. A. Douillard, M. Cord, C. Ollion, T. Robert and E. Valle, Podnet: Pooled outputs distillation for small-tasks incremental learning, in *Computer vision–ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part XX 16*, (Springer International Publishing, 2020), pp. 86–102.
40. S. Yan, J. Xie and X. He, Der: Dynamically expandable representation for class incremental learning, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 3014–3023.
41. E. Chee, M. L. Lee and W. Hsu, Leveraging old knowledge to continually learn new classes in medical images, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** June 2023, pp. 14178–14186.
42. F. Zhu, X.-Y. Zhang, C. Wang, F. Yin and C.-L. Liu, Prototype augmentation and self-supervision for incremental learning, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5871–5880.
43. R. Aljundi, M. Lin, B. Goujaud and Y. Bengio, Gradient based sample selection for online continual learning, *Advances in neural information processing systems* **32** (2019).
44. A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr and M. Ranzato, On tiny episodic memories in continual learning, *arXiv preprint arXiv:1902.10486* (2019).
45. R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin and L. PageCaccia, Online continual learning with maximal interfered retrieval, in *Advances in Neural Information Processing Systems*, **32**, (IEEE, 2019).
46. A. Prabhu, P. H. Torr and P. K. Dokania, Gdumb: A simple approach that questions our progress in continual learning, in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, (Springer, 2020), pp. 524–540.
47. D. Shim, Z. Mai, J. Jeong, S. Sanner, H. Kim and J. Jang, Online class-incremental continual learning with adversarial shapley value, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **35**(11), (AAAI Press, 2021), pp. 9630–9638.
48. Z. Mai, R. Li, H. Kim and S. Sanner, Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning, in *Proceedings of the IEEE/CVF*

- Conference on Computer Vision and Pattern Recognition*, CVPR2021, pp. 3589–3599.
49. Y. Gu, X. Yang, K. Wei and C. Deng, Not just selection, but exploration: Online class-incremental continual learning via dual view consistency, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7442–7451.

DAKTA: Directional Kolmogorov-Arnold Classifier for Task Arithmetic in Continual Learning

Quynh-Trang Pham Thi¹[0009-0007-4151-6467], Van-Truong Le^[0009-0002-3647-5860]², Thanh-Huyen Pham^[0000-0001-7458-6655]³, Thanh-Nga Hoang Thi^[0009-0004-0520-6451]¹, and Duc-Trong Le^[0000-0003-4621-8956]^{1*}

¹ Vietnam National University, University of Engineering and Technology,
Hanoi, Vietnam

{trangptq,23021647, trongld}@vnu.edu.vn

² University of Information Technology,
Ho Chi Minh, Vietnam

truong.lvt.24@gmail.com

³ Halong university, Quang Ninh, Vietnam {phamthanhhuyn@daihochalong.edu.vn}

Abstract. Continual learning requires models to acquire new knowledge while preserving previously learned information, which is a fundamental challenge known as the stability-plasticity tradeoff. While recent advances in model compositionality through task arithmetic show promise, existing approaches primarily rely on linear classifiers that struggle to maintain stable classification spaces during incremental learning. In this paper, we propose Directional Arithmetic Kolmogorov Task Architecture (DAKTA), a strategy that addresses both compositionality and classification stability in continual learning. DAKTA leverages second-order Taylor approximation theory to ensure task vectors remain within the pre-training basin, enabling effective model composition for enhanced stability. Our key innovation is to replace conventional linear classifiers with learnable Gaussian Radial Basis Functions (RBFs) that provide selective channel activation, enhanced locality, and significantly reduced interference between tasks, thereby boosting plasticity. Furthermore, we proposed a novel directional logit fusion mechanism that combines RBF-based magnitude information with feature directional cues, enabling the classifier to capture both local similarity patterns and global directional relationships for more robust class discrimination. Experiments on ImageNet-R, CUB-200, and CIFAR-100 demonstrate significant improvements over state-of-the-art methods with DAKTA.

Keywords: Continual Learning · Task Arithmetic · Directional Kolmogorov-Arnold.

* Corresponding author: trongld@vnu.edu.vn

1 Introduction

Continual learning (CL) enables models to learn new tasks sequentially while retaining performance on previous ones, a critical capability for real-world applications. The stability-plasticity dilemma of continual learning poses a significant challenge, where plasticity refers to the ability to acquire new knowledge, and stability retains the knowledge of previous tasks, particularly in the Class Incremental Learning (CIL) scenario [5]. Many methods tackle this problem by storing previous samples, which are known as replay-based methods[8, 4, 2], although some applications are legally prohibited from retaining training data from previous tasks. Architecture-based techniques[3] is another approach resolve the stability-plasticity tradeoff via dynamic parameter adjustments, where each task will learn specific parameter regions, thereby enabling the model to learn well on new tasks without affecting old tasks without affecting old tasks without needing to store old data. Regularization-based approaches restrict parameter modifications to maintain knowledge from earlier tasks[7, 1]. In this paper, we propose DAKTA, which combines the advantages of 2 approaches: architecture-based and regularization. Conventional architecture-based methods typically expand linear classifier heads, which struggle to maintain a stable classification space while learning new tasks. A new architecture called Kolmogorov–Arnold Networks (KAN)[6] has recently emerged, showing inherent advantages for continual learning scenarios. Inspired by the success of KAN in preserving learning stability during simple continual regression tasks, we expand the classifier head models based on the KAN architecture into ViT(Vision Transformer) to leverage the power of a pre-trained model. Additionally, when fine-tuning the model on new tasks, we will use loss regularization to stay within the pre-training basin to achieve composable modules.

DAKTA replaces linear classifiers with Gaussian Radial Basis Functions (RBFs) within a Low-Rank Adaptation (LoRA)-tuned Vision Transformer (ViT), enabling selective feature activation to mitigate interference. Our model penalize the importance parameters grounded in second-order Taylor approximations, composes task-specific parameters (task vectors) while regularizing to stay within the pre-training basin. Our model excels in fine-grained classification and task compositionality; their integration remains unexplored by selectively activating features and ensuring parameter stability. Our contributions are as follows:

- A unified CL framework integrating an RBF-based classifier with task arithmetic.
- An adapted Fisher computation for parameters of the classifier head, ensuring compatibility with regularization.
- Extensive experiments on ImageNet-R, CUB200, and CIFAR-100, demonstrating superior accuracy and reduced forgetting.

2 Methodology

DAKTA is designed to train each task on each composed model, and is inspired by the Kolmogorov-Arnold representation theorem that combines many fundamental functions for learnable activation functions on the edges of the classifier. Additionally, DAKTA uses the second-order regularization loss, which helps the model protect important knowledge from pre-trained models built upon a Vision Transformer (ViT) backbone adapted for each new task using Low-Rank Adaptation (LoRA). The following subsections detail each component and the overall learning process.

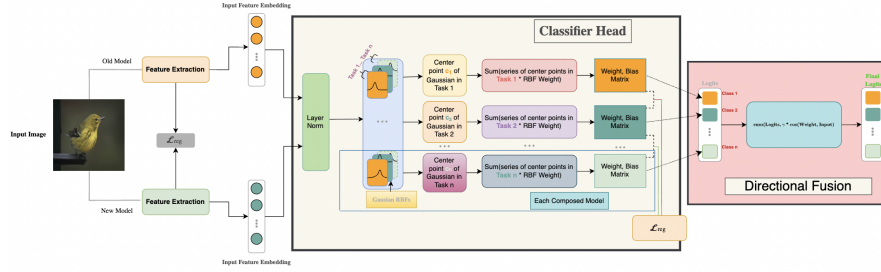


Fig. 1. Pipeline of the DAKTA framework, integrating a Vision Transformer (ViT) backbone with Low-Rank Adaptation (LoRA) and a Kolmogorov-Arnold Classifier (KAC) using Gaussian Radial Basis Functions (RBFs) for class-incremental learning. The process involves feature extraction from both old and new models, task-specific parameter composition via Incremental Task Arithmetic (ITA), and regularization using the Fisher Information Matrix to mitigate catastrophic forgetting.

2.1 Vision Transformer Backbone and Low-Rank Adaptation (LoRA)

Our framework uses a Vision Transformer (ViT) as the feature extractor. The ViT is a neural network architecture that processes images by dividing them into patches and applying self-attention mechanisms, allowing it to capture complex patterns in visual data. To efficiently adapt the ViT to new tasks without retraining all its parameters, we use *Low-Rank Adaptation* (LoRA). LoRA works by inserting small, trainable matrices (of low rank) into the attention and feed-forward layers of the ViT. During training on each new task, only these LoRA parameters are updated, while the majority of the ViT's parameters remain fixed. This approach enables fast adaptation to new tasks and helps prevent the model from overwriting previously learned knowledge.

2.2 Incremental Task Arithmetic (ITA) and Model Composition

To enable the model to learn new tasks without forgetting previous ones, we use *Incremental Task Arithmetic* (ITA). For each task t , we keep track of the changes made to both the LoRA parameters of the ViT and the parameters of the KAC classifier. Let θ_0 and ψ_0 denote the initial parameters of the ViT and KAC, respectively, before any task-specific adaptation. After training on task t , the updated parameters are θ_t and ψ_t . We define the *task vectors* as:

$$\tau_t^{\text{backbone}} = \theta_t - \theta_0, \quad \tau_t^{\text{cls}} = \psi_t - \psi_0. \quad (1)$$

At inference time (i.e., when making predictions after all tasks have been learned), we combine the knowledge from all T tasks by averaging these task vectors:

$$\theta_P = \theta_0 + \frac{1}{T} \sum_{t=1}^T \tau_t^{\text{backbone}}, \quad \psi_P = \psi_0 + \frac{1}{T} \sum_{t=1}^T \tau_t^{\text{cls}}. \quad (2)$$

This means that the final model is a balanced combination of all the adaptations made for each task, which helps retain knowledge from earlier tasks while still incorporating new information.

2.3 Kolmogorov-Arnold Classifier Head

We introduce the Kolmogorov-Arnold Classifier (KAC), which is more flexible than a standard linear classifier. The KAC is inspired by a mathematical result called the Kolmogorov-Arnold representation theorem, which states that any complex function can be constructed from sums and compositions of simple, one-dimensional functions. In our implementation, the KAC uses *Gaussian Radial Basis Functions* (RBFs) to model the relationship between features and classes.

Given a feature vector $x \in \mathbb{R}^d$ (where d is the number of features), the KAC computes, for each class c , an activation as follows:

$$\phi_c(x) = \sum_{j=1}^N w_{c,j} \exp \left(-\frac{(x - \mu_{c,j})^2}{2\sigma_{c,j}^2} \right), \quad (3)$$

where:

- N is the number of RBFs per feature channel (a small integer, e.g., 3 or 4),
- $w_{c,j}$ are learnable weights for class c and RBF j ,
- $\mu_{c,j}$ are learnable centers (the location where each RBF is focused),
- $\sigma_{c,j}$ are learnable widths (which control how wide each RBF is),
- $\exp(-\frac{(x - \mu_{c,j})^2}{2\sigma_{c,j}^2})$ is the Gaussian RBF, which outputs a value close to 1 when x is near $\mu_{c,j}$ and quickly drops to 0 as x moves away.

Given a feature vector $x \in \mathbb{R}^d$ (where d is the number of features), the KAC constructs an effective weight matrix for each class c by combining the RBF

centers and weights: $W_c = \sum_{j=1}^N w_{c,j} \mu_{c,j}$, where $w_{c,j}$ are learnable weights, $\mu_{c,j}$ are learnable centers, and N is the number of RBFs per feature channel. The initial logits are computed via a linear transformation:

$$l_c^{\text{init}}(x) = W_c \cdot x + b_c, \quad (4)$$

where b_c is a learnable bias term for class c . To enhance the classifier’s expressiveness, a directional alignment term is computed as the cosine similarity between the normalized feature vector $\frac{x}{\|x\|_2}$ and the normalized effective weight vector $\frac{W_c}{\|W_c\|_2}$, defined as $\cos(x, W_c) = \frac{x \cdot W_c}{\|x\|_2 \|W_c\|_2}$.

2.4 Directional Vector Fusion

In addition to utilizing vector magnitude information, we also integrate directional information into our approach. The final logits are computed by adding this term, scaled by a learnable parameter γ_c initialized to 0.1, to the initial logits:

$$l_c(x) = l_c^{\text{init}}(x) + \gamma_c \cdot \cos(x, W_c). \quad (5)$$

This design combines the linear transformation with a directional component, where the learnable scaling parameter γ_c optimizes the contribution of the cosine similarity term, improving robustness and expressiveness compared to a standard linear classifier.

2.5 Second-Order Regularization with the Fisher Information Matrix

To further reduce forgetting, we use a regularization technique based on the *Fisher Information Matrix* (FIM). The FIM is a mathematical tool that measures how important each parameter is for the model’s predictions. After training on each task, we estimate the diagonal of the FIM for both the ViT and the KAC parameters, denoted as $\hat{F}_{\theta_0}$ and $\hat{F}_{\psi_0}$. During training on a new task, we penalize large changes to parameters that the FIM indicates are important for previous tasks. The regularization terms are as follows:

$$L_{\text{reg}}^{\text{backbone}} = \frac{\alpha}{2} \sum_i \hat{F}_{\theta_0}^{(i)} (\theta_t^{(i)} - \theta_0^{(i)})^2, \quad (6)$$

$$L_{\text{reg}}^{\text{cls}} = \frac{\beta}{2} \sum_j \hat{F}_{\psi_0}^{(j)} (\psi_t^{(j)} - \psi_0^{(j)})^2, \quad (7)$$

where α and β are hyperparameters that control how strongly we penalize changes to important parameters. This approach encourages the model to keep parameters that are crucial for previous tasks close to their original values, thus preserving old knowledge.

2.6 Training Objective and Step-by-Step Procedure

For each batch of data (x, y) in task t , the total loss is:

$$L = L_{\text{CE}}(h_{\psi_t}(f_{\theta_t}(x)), y) + L_{\text{reg}}^{\text{backbone}} + L_{\text{reg}}^{\text{cls}}, \quad (8)$$

where L_{CE} is the cross-entropy loss, which measures how well the model’s predictions match the true labels, f_{θ_t} is the ViT with LoRA parameters for task t , h_{ψ_t} is the KAC classifier for task t , $L_{\text{reg}}^{\text{backbone}}$ and $L_{\text{reg}}^{\text{cls}}$ are the regularization terms described above.

The training procedure for each task is as follows Algorithm 1.

Algorithm 1: DAKTA Algorithm

Input: Pre-trained Vision Transformer (ViT) backbone parameters θ_0 ;
initial KAC classifier parameters ψ_0
Output: Final composed model parameters (θ_P, ψ_P)
for each task $t = 1, \dots, T$ **do**
 Classifier Expansion: Extend KAC classifier to accommodate new
 classes for task t ;
 Parameter Initialization: $\theta_t \leftarrow \theta_0, \psi_t \leftarrow \psi_0$;
 for each training iteration on task t **do**
 Data Sampling: Sample a mini-batch of data (x, y) from task t ;
 Feature Extraction: Extract features $z = f_{\theta_t}(x)$ using
 ViT+LoRA backbone;
 Logit Computation: Compute logits $l = h_{\psi_t}(z)$ using KAC
 classifier;
 Classification Loss: Compute task loss
 $L_{\text{task}} = \text{CrossEntropy}(l, y)$;
 Regularization Losses: Compute second-order regularization
 terms: (Eq. 6 and Eq. 7) **Compute Total Loss** **8**
 Parameter Update: Update θ_t, ψ_t via gradient descent on L ;
 end
 Fisher Information Estimation: Estimate $\hat{F}_{\theta_0}$ and $\hat{F}_{\psi_0}$ using
 data from task t ;
 Task Vector Storage: Store task vectors:

$$\tau_t^{\text{backbone}} = \theta_t - \theta_0 \quad (9)$$

$$\tau_t^{\text{cls}} = \psi_t - \psi_0 \quad (10)$$

end
Directional vector fusion: Compose final model parameters following
Eq. 5

After all tasks have been learned, the final model is composed of averaging the task vectors as described above. This ensures that the model maintains a balance between retaining knowledge from earlier tasks and adapting to new ones, achieving robustness in continual learning without catastrophic forgetting.

2.7 Algorithm

We introduce **DAKTA**, a continual learning algorithm that unifies parameter-efficient adaptation and a flexible classifier head for class-incremental scenarios. As detailed in Algorithm 1, DAKTA operates in two main phases for each new task: classifier expansion and regularized fine-tuning.

Classifier Expansion. When a new task arrives, the KAC classifier is expanded to accommodate the additional classes. To ensure stable integration of new knowledge, we initialize the task-specific parameters for both the backbone and classifier from their pre-trained values. In this stage, we optionally perform a brief linear probing step, where only the new classifier parameters are updated while the backbone remains frozen. This step helps align the new head with the pre-trained feature space, promoting optimal initialization for subsequent learning.

Regularized Fine-Tuning. After pre-consolidation, DAKTA proceeds to jointly update the backbone (via LoRA modules) and the KAC classifier using the data from the current task. The training objective combines the standard cross-entropy loss with second-order regularization terms for both the backbone and classifier parameters. These regularizers, based on the Fisher information estimated after each task, penalize deviations from the pre-trained weights, thereby mitigating catastrophic forgetting. Throughout training, only a small subset of parameters (the LoRA adapters and the KAC head) are updated, ensuring parameter efficiency and scalability.

Task Vector Storage and Model Composition. Upon completion of each task, DAKTA records the parameter changes (task vectors) for both the backbone and the classifier. The Fisher information matrices are also updated to reflect the importance of each parameter. At inference, the final model is composed by aggregating the pre-trained parameters with a weighted sum of the task vectors, balancing knowledge across all tasks.

DAKTA is compatible with various parameter-efficient adaptation strategies, such as LoRA and (IA)³, and is designed to be robust to the continual introduction of new classes. By combining a compositional classifier with principled regularization and efficient adaptation, DAKTA achieves strong performance and knowledge retention in class-incremental learning.

3 Experimental and Result

We evaluate our proposed method on a diverse set of continual learning benchmarks, including ImageNet-R (IN-R), CIFAR-100 (C-100), CUB-200 (CUB), Caltech-256 (Caltech), MIT-67 (MIT), RESISC-45 (RESISC), and CropDiseases (CropDis.). Our experiments compare the final classification accuracy of our approach with that of a range of state-of-the-art (SOTA) continual learning methods, including both fine-tuning and prompt-based strategies. The compared baselines include CODA, SEED, TMC, DER++, and InfLoRA, as well as several ITA-based variants (ITA-FFT, ITA-LoRA, ITA-(IA)³). Each method

is evaluated under the same experimental protocol, with results reported as the final accuracy (%) after learning all tasks. For methods that utilize experience replay, such as DER++, a buffer size of 1,000 examples is used. Prompt-based methods (L2P, CODA, APT) and LoRA-based fine-tuning (InfLoRA) are also included for comprehensive comparison. Table 1 summarizes the results across all datasets. Our method achieves the highest accuracy on the majority of benchmarks, demonstrating the effectiveness of combining ITA-LoRA with the Kolmogorov-Arnold Classifier (KAC) for continual learning. The best results for each dataset are highlighted in bold.

Table 1. Comparison with SOTA (Final Accuracy [↑]). Best results in bold. EWC, LwF-MC, DER++ (buffer size of 1000 examples), SEED, and TMC rely on full fine-tuning; L2P, CODA, and APT utilize prompt-based learning. Finally, InfLoRA adopts LoRA fine-tuning.

Model	C-100	CUB	Caltech	MIT	RESISC	CropDis.
CODA [10]	86.48	78.54	90.57	77.73	69.50	74.65
SEED [11]	83.39	85.35	90.04	86.34	74.81	92.77
TMC [14]	78.42	71.72	82.30	68.66	60.66	66.56
DER++ [9]	83.02	76.10	86.11	68.88	67.23	98.77
InfLoRA [12]	87.17	79.14	90.53	79.14	79.92	89.05
ITA-FFT [15]	89.38	84.80	92.32	85.35	80.50	91.81
ITA-LoRA [15]	89.96	85.55	92.65	86.60	82.00	95.85
ITA-(IA)3 [15]	90.66	85.67	92.67	84.74	83.73	95.41
DAKTA (Ours)	91.21	87.59	93.61	87.97	85.82	97.75

To assess the performance of our model in a continual learning scenario, we also evaluate its accuracy across multiple datasets and visualize the results to highlight the impact of catastrophic forgetting. Figure 2 presents a comprehensive comparison of the model’s accuracy as it learns new tasks sequentially, revealing trends in performance retention and degradation over time.

4 Ablation Study: Effect of Second-Order Regularization Placement

To investigate the impact of second-order (Fisher-based) regularization in our framework, we conduct an ablation study focusing on where the regularization loss is applied. Specifically, we compare two settings:

1. **Backbone-only Regularization:** The second-order regularization loss is applied only to the ViT backbone parameters, while the KAC classifier head is trained without any regularization.
2. **Backbone + Classifier Regularization:** The second-order regularization loss is applied to both the ViT backbone and the KAC classifier head parameters.

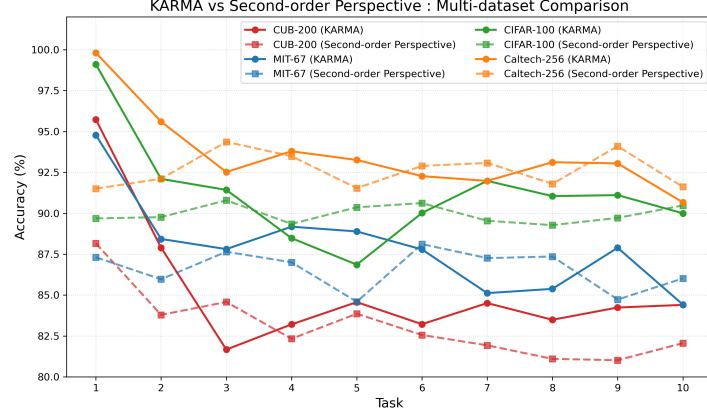


Fig. 2. Comparison of model accuracy across multiple datasets in a continual learning setting, highlighting the impact of catastrophic forgetting.

We evaluate both settings on the class-incremental learning benchmark and report the average accuracy (%) over all tasks and the accuracy on the last task, which is a common measure of resistance to catastrophic forgetting.

Table 2. Ablation study on second-order regularization.

Method	C-100	CUB	Cal.	MIT	RES.	Crop.
w/o Reg.	85.24	80.57	89.1	82.25	79.38	90.84
w/o KAC	89.91	84.12	92.35	85.24	82.25	94.82
Full Model	91.21	87.59	93.61	87.97	85.82	97.75

As shown in Table 3, applying regularization (shown in Equations (6) and (7)) to both the backbone and the KAC classifier head leads to a clear improvement in the accuracy of each task. This demonstrates that regularizing the classifier head, in addition to the backbone, further mitigates catastrophic forgetting and enhances overall continual learning performance.

5 Conclusion

In this paper, we presented DAKTA (Directional Arithmetic Kolmogorov Task Architecture), a novel framework that successfully addresses the stability-plasticity tradeoff in continual learning. By integrating Incremental Task Arithmetic (ITA) with Kolmogorov-Arnold Classifiers (KAC), DAKTA achieves superior performance in class-incremental learning scenarios. Our key innovation lies in replacing conventional linear classifiers with learnable Gaussian Radial Basis Func-

Table 3. Comparison with the SOTA across several benchmarks(Final Forgetting ($\downarrow$)). Note that a lower forgetting score indicates better performance.

Model	C-100	CUB	Caltech	MIT	RESISC	CropDis.
CODA [10]	5.28	8.82	2.64	7.49	22	15.37
SEED [11]	6.39	5.12	2.67	4.53	8.66	1.19
InfLoRA [12]	4.05	4.58	1.89	6.85	10.77	4.79
APT [13]	6.71	8.82	3.64	6.33	27.74	14.01
TMC [14]	9.2	7.37	6.39	12.43	16.43	16.94
ITA-LoRA [15]	3.22	2.08	2.38	4.68	8.30	0.68
ITA-FFT [15]	3.89	1.39	2.44	4.80	6.19	0.95
ITA-(IA)3 [15]	3.57	2.23	2.39	4.48	7.33	0.76
DAKTA(Ours)	2.98	1.14	1.93	3.08	4.38	0.75

tions, which provide enhanced locality properties and significantly reduce interference between tasks. The directional logit fusion mechanism further strengthens classification by combining RBF-based magnitude information with feature directional cues. Experimental evaluation across six diverse benchmarks consistently demonstrates that DAKTA outperforms state-of-the-art continual learning methods, achieving the highest final accuracy while maintaining the lowest forgetting scores. Ablation studies confirm the critical importance of applying Fisher Information-based regularization to both backbone and classifier components.

References

1. Chen, X.; and Chang, X. 2023. Dynamic residual classifier for class incremental learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 18743–18752.
2. Garg, S.; Farajtabar, M.; Pouransari, H.; Vemulapalli, R.; Mehta, S.; Tuzel, O.; Shankar, V.; and Faghri, F. 2023. Tic-clip: Continual training of clip models. arXiv preprint arXiv:2310.16226.
3. Kim, T.; Park, J.; and Han, B. 2024. Cross-class feature augmentation for class incremental learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, 13168–13176.
4. Li, Y.; Li, Q.; Wang, H.; Li, R.; Zhong, W.; and Zhang, G. 2024. Towards efficient replay in federated incremental learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 12820–12829.
5. Lin, G.; Chu, H.; and Lai, H. 2022. Towards better plasticity-stability trade-off in incremental learning: A simple linear connector. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 89–98.
6. Liu, Z.; Wang, Y.; Vaidya, S.; Ruehle, F.; Halverson, J.; Soljačić, M.; Hou, T. Y.; and Tegmark, M. 2024. Kan: Kolmogorov-arnold networks. arXiv preprint arXiv:2404.19756.

7. Magistri, S.; Trinci, T.; Soutif-Cormerais, A.; van de Weijer, J.; and Bagdanov, A. D. 2024. Elastic feature consolidation for cold start exemplar-free incremental learning. arXiv preprint arXiv:2402.03917
8. Tong, R.; Liu, Y.; Shi, J. Q.; and Gong, D. 2025. Coreset selection via reducible loss in continual learning. In *The Thirteenth International Conference on Learning Representations*.
9. Buzzega, P., Boschini, M., Porrello, A., Abati, D., & Calderara, S. (2020). Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33, 15920-15930.
10. Smith, J. S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., ... & Kira, Z. (2023). Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11909-11919).
11. Rypešć, G., Cygert, S., Khan, V., Trzciński, T., Zieliński, B., & Twardowski, B. (2024). Divide and not forget: Ensemble of selectively trained experts in continual learning. arXiv preprint arXiv:2401.10191
12. Liang, Y. S., & Li, W. J. (2024). Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 23638-23647).
13. Bowman, B., Achille, A., Zancato, L., Trager, M., Perera, P., Paolini, G., & Soatto, S. (2023). a-la-carte prompt tuning (apt): Combining distinct data via composable prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14984-14993).
14. Liu, T. Y., & Soatto, S. (2023). Tangent model composition for ensembling and continual fine-tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 18676-18686).
15. Porrello, A., Bonicelli, L., Buzzega, P., Millunzi, M., Calderara, S., & Cucchiara, R. (2024). A second-order perspective on model compositionality and incremental learning. arXiv preprint arXiv:2405.16350

SO-LoRA: Sparse Orthogonal LoRA for Parameter-Efficient Continual Learning

Anonymous ICME submission

Abstract—Applying Low-Rank Adaptation (LoRA) to Continual Learning presents a structural dilemma: instantiating task-specific adapters leads to unsustainable linear parameter growth, while sharing adapters risks catastrophic forgetting due to gradient interference. In this work, we introduce Sparse Orthogonal LoRA (SO-LoRA), a novel framework that resolves this trade-off by reconceptualizing adaptation as the learning of sparse, evolving compositions over a fixed, universal orthogonal basis. Rather than isolating parameters or naively sharing them, SO-LoRA constrains all task-specific knowledge to a unified, stable subspace. This is achieved through a dual mechanism: group sparsity regularization minimizes interference by enforcing highly compact task representations, while a gradient-aware soft mask dynamically shields critical compositions to preserve accumulated knowledge. We analyze and demonstrate that our sparsity constraints explicitly bound the interference between task updates. Empirically, SO-LoRA outperforms existing methods across diverse benchmarks, particularly in long-sequence tasks, without increasing parameter complexity.

Index Terms—Continual learning, LoRA, Sparsity, Orthogonality, Gradient Interference

I. INTRODUCTION

Pre-trained models (PTMs) have shifted continual learning (CL) from representation learning without prior knowledge to *adapting* strong, general-purpose features across a stream of tasks [1], [2]. In practice, this shift is enabled by parameter-efficient fine-tuning (PEFT), where Low-Rank Adaptation (LoRA) is a widely adopted choice [3]. Yet, while LoRA is effective for single-task adaptation, applying it in sequential settings exposes a structural tension.

Existing LoRA-based CL methods largely choose between two imperfect options. Architecture-based isolation allocates a separate low-rank adapter per task [4], eliminating interference but incurring linear growth in parameters and storage. Shared-adapter training keeps memory constant, but is vulnerable to catastrophic interference: gradients from different tasks compete within the same low-rank subspace and overwrite previously acquired knowledge [5], [6]. Previous attempts to reduce interference often rely on heuristic controls (e.g., shrinking updates or unlearning) [7], [8], which mitigate conflict without directly addressing the underlying cause of task representations overlapping in the adaptation subspace.

We propose that the core issue is geometric and the remedy should be geometric as well. We introduce **Sparse Orthogonal LoRA (SO-LoRA)**, which replaces parameter isolation with *representational orthogonality*. SO-LoRA fixes a universal orthogonal basis to prevent drift and learns task-specific adaptations as sparse compositions within this stable coordinate system. Group sparsity encourages each task to activate only

a small subset of basis directions, reducing overlap across tasks, while a gradient-aware soft mask protects historically important directions from being overwritten.

We provide a theoretical analysis that yields an explicit upper bound on interference as a function of row-wise sparsity. Empirically, SO-LoRA scales favorably on long task sequences, outperforming strong baselines on ImageNet-R and ImageNet-A, especially in the 20-task regime, while maintaining a constant parameter budget comparable to a single LoRA adapter.

Our main contributions are, as follows:

- **Method:** SO-LoRA, a continual adaptation framework that models task updates as sparse compositions on a fixed orthogonal basis.
- **Theory:** An interference bound linking the sparsity of the group to reduced task overlap, motivating an objective that combines $\ell_{2,1}$ regularization with gradient-aware masking.
- **Empirics:** Strong rehearsal-free results across benchmarks, with notably improved retention in long-horizon sequences under constant parameter cost.

II. RELATED WORK

To motivate the proposed structural constraints, we briefly review the progression of Continual Learning approaches. We focus on how adaptation strategies for pre-trained models expose a fundamental tension between stability and efficiency in parameter-efficient methods—a tension that SO-LoRA is explicitly designed to address.

A. Continual Learning with Pre-Trained Models

Most rehearsal-free PTM-based CL methods fall into *prompting* and *adapters/PEFT*. Prompting methods (e.g., L2P, DualPrompt) learn task-conditioned tokens on top of a frozen backbone [9], [10], and later variants use attention to select prompts dynamically (CODA-Prompt, HiDe-Prompt) [11], [12]. While effective, prompting modifies the input interface rather than the internal transformation, which can be less robust under distribution shifts and may introduce inference overhead due to longer token sequences. This has shifted recent attention toward PEFT modules that directly adjust internal representations.

B. Low-Rank Adaptation in Continual Learning

LoRA is a widely adopted PEFT technique [3], but sequential learning exposes a central tension between *stability* and *scalability*. Architecture-based isolation instantiates per-task

LoRA modules [4], [13], eliminating interference but incurring linear parameter growth and limiting beneficial cross-task transfer. At the other extreme, sharing a single LoRA across tasks keeps memory constant but is prone to gradient interference: unrelated tasks update the same low-rank subspace and overwrite prior knowledge [5], [6]. Recent methods (e.g., SD-LoRA, LoRA-DRS) reduce forgetting via optimization heuristics such as decoupling or unlearning [7], [8], yet they largely treat interference as an optimization artifact rather than a geometric consequence of overlapping task representations.

C. Orthogonality and Compositional Subspaces

To address interference structurally, several works impose orthogonality or interference-free parameterizations. O-LoRA encourages orthogonal subspaces but depends on regularizers that may drift or become costly over long sequences [14]. InfLoRA enforces a more rigid interference-free form by reparameterizing the adaptation matrix [15], but still treats the adapter as a single constrained object rather than a reusable set of primitives. In parallel, modular approaches such as AdapterFusion and LoRAHub combine multiple experts to compose solutions [16], [17], though typically at a coarse granularity (whole adapters), which can be memory-inefficient. **Our Contribution.** SO-LoRA connects these directions by making the adaptation space both *stable* and *compositional*. We fix an orthonormal basis to prevent representational drift and learn task updates as *sparse* combinations over this shared dictionary. This yields fine-grained reuse (vector-level composition) while maintaining a constant footprint, capturing the benefits of isolation without linear growth and reducing interference by construction.

III. PROPOSED METHOD

We propose **SO-LoRA**, a rehearsal-free continual learning scheme that encodes each task as a *sparse* composition over a *fixed orthogonal* LoRA basis. The key idea is to keep the low-rank coordinate system stable over time, and let tasks compete only through a sparse set of shared coefficients.

A. Preliminary

Problem setting. We study rehearsal-free Class-Incremental Learning (CIL). A model f_Φ receives a stream of T tasks $\{\mathcal{T}_1, \dots, \mathcal{T}_T\}$, where each task $\mathcal{T}_t$ provides data $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t} \sim P_t(x, y)$ and has a disjoint label set C_t ($C_t \cap C_k = \emptyset$ for $t \neq k$). We adopt parameter-efficient fine-tuning by splitting parameters as $\Phi = \{\Theta, \Psi\}$, with a frozen backbone Θ and a small trainable adapter Ψ ($|\Psi| \ll |\Theta|$). The ideal objective minimizes risk over all tasks:

$$\Phi^* = \operatorname{argmin}_{\Phi} \sum_{t=1}^T \mathbb{E}_{(x,y) \sim \mathcal{D}_t} [\mathcal{L}(f_\Phi(x), y)]. \quad (1)$$

In rehearsal-free CIL, however, task $\mathcal{T}_t$ is optimized using only $\mathcal{D}_t$; gradients $\nabla_\Psi \mathcal{L}_t$ can therefore move parameters to regions that increase loss on earlier tasks (catastrophic forgetting).

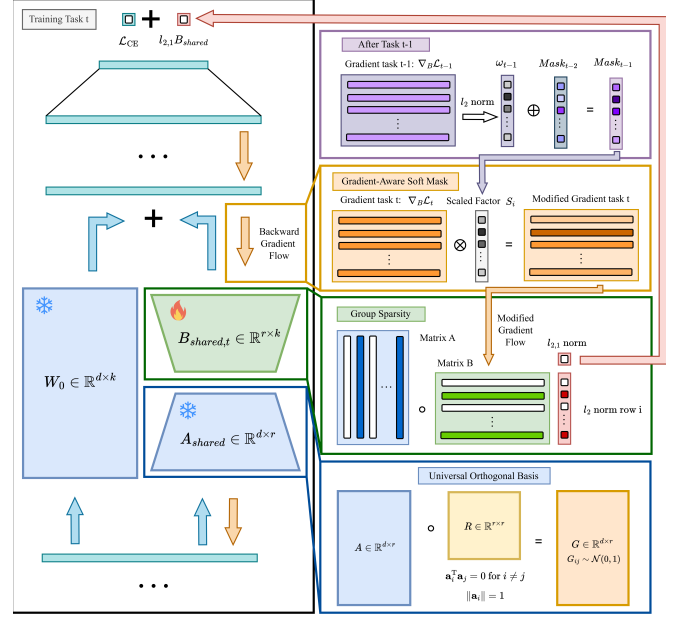


Fig. 1: SO-LoRA represents adaptation as sparse coefficients $B^{(t)}$ over a frozen orthonormal basis A . Group sparsity limits overlap across tasks, while a gradient-aware mask M protects historically important rank dimensions.

LoRA in continual learning. LoRA [3] adapts a frozen weight $W_0 \in \mathbb{R}^{d \times k}$ using a low-rank update $\Delta W = AB$, with $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times k}$:

$$h = xW_0 + x\Delta W = xW_0 + xAB. \quad (2)$$

In a task stream, one can either *isolate* adapters per task (stable but memory grows with T):

$$W^{(t)} = W_0 + A_t B_t, \quad \Omega_{\text{iso}} = \bigcup_{t=1}^T \{A_t, B_t\} \Rightarrow |\Omega_{\text{iso}}| \propto \mathcal{O}(T), \quad (3)$$

or *share* a single adapter (constant memory but prone to interference):

$$W^{(t)} = W_0 + A^{(t)} B^{(t)}, \quad \Omega_{\text{shared}} = \{A, B\} \Rightarrow |\Omega_{\text{shared}}| \propto \mathcal{O}(1), \quad (4)$$

often yielding

$$\mathcal{L}_{<t}(W_0 + A^{(t)} B^{(t)}) \gg \mathcal{L}_{<t}(W_0 + A^{(t-1)} B^{(t-1)}). \quad (5)$$

SO-LoRA targets this trade-off: it keeps $\mathcal{O}(1)$ parameters while enforcing subspace separation within the shared adapter.

B. SO-LoRA: Sparse Orthogonal LoRA

Given frozen pre-trained weights W_0 , SO-LoRA parameterizes the task-adapted weights as

$$W^{(t)} = W_0 + AB^{(t)}, \quad (6)$$

where $A \in \mathbb{R}^{d \times r}$ is a *fixed* orthogonal basis and $B^{(t)} \in \mathbb{R}^{r \times k}$ is the *shared, evolving* coefficient matrix. Relative to standard

LoRA (where both factors drift), Eq. 6 freezes the low-rank coordinate system and restricts learning to a controlled coefficient space.

1) *Representation Stability via Fixed Orthogonal Basis:*

A major source of forgetting in low-rank adaptation is *subspace rotation*: when both factors are updated, the meaning of rank dimensions changes over time. We prevent this by constructing A once and freezing it. Concretely, we sample $G \sim \mathcal{N}(0, 1)^{d \times r}$ and perform a QR decomposition $G = QR$, setting $A \leftarrow Q$. This yields

$$A^\top A = I_r \Rightarrow \langle A_{:,i}, A_{:,j} \rangle = \delta_{ij}. \quad (7)$$

With a fixed orthonormal dictionary, tasks can only differ through $B^{(t)}$, which reduces representational drift and makes interference analyzable in the coefficient space.

2) *Interference Minimization via Group Sparsity:* With A fixed, interference arises when multiple tasks activate the same basis dimensions. Since $AB^{(t)} = \sum_{j=1}^r A_{:,j} B_{j,:}^{(t)}$, the row $B_{j,:}^{(t)}$ controls whether basis vector $A_{:,j}$ is used. We encourage row-wise sparsity with the $\ell_{2,1}$ norm (group lasso):

$$\mathcal{R}(B^{(t)}) = \|B^{(t)}\|_{2,1} = \sum_{j=1}^r \|B_{j,:}^{(t)}\|_2 = \sum_{j=1}^r \sqrt{\sum_{i=1}^k (B_{ji}^{(t)})^2}. \quad (8)$$

The per-task objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}}(f_\Phi(x), y) + \lambda \sum_{j=1}^r \|B_{j,:}^{(t)}\|_2. \quad (9)$$

This pushes each task to reuse as few rank directions as necessary, reducing subspace overlap and thus expected interference.

3) *Protecting important directions with a gradient-aware mask:* Sparsity lowers collision probability, but cannot guarantee that critical directions will not be overwritten. We therefore maintain an importance vector $M \in \mathbb{R}^r$, updated only at task boundaries. After finishing task $\mathcal{T}_t$, we measure the sensitivity of each rank dimension by the gradient norm of the corresponding row:

$$\omega_{t,j} = \|(\nabla_{B^{(t)}} \mathcal{L}_t)_{j,:}\|_2. \quad (10)$$

We aggregate importance over time using an EMA:

$$M_{t,j} = \alpha \cdot \frac{\omega_{t,j}}{\max_k(\omega_{t,k}) + \epsilon} + (1 - \alpha) \cdot M_{t-1,j}. \quad (11)$$

During training of $\mathcal{T}_{t+1}$, we compute batch-wise relevance $\phi_j = \frac{\|g_{j,:}\|_2}{\max_k \|g_{k,:}\|_2 + \epsilon}$ for $g = \nabla_{B^{(t+1)}} \mathcal{L}_{t+1}$, and scale gradients by

$$S_j = 1 - (M_{t,j}(1 - \phi_j)), \quad g'_{j,:} = S_j \cdot g_{j,:}. \quad (12)$$

Thus, historically important but currently irrelevant rows ($M_{t,j} \approx 1, \phi_j \approx 0$) are suppressed, while unused capacity ($M_{t,j} \approx 0$) remains free to adapt.

C. Theoretical Analysis

We formalize interference through subspace overlap. For task $\mathcal{T}$, define the effective update $\Delta W = A\Delta B$. Interference between tasks i and j is captured by the Frobenius inner product $\langle \Delta W_i, \Delta W_j \rangle_F$.

Proposition 1. Let $\Delta W_i = A\Delta B_i$ and $\Delta W_j = A\Delta B_j$ for two tasks, with $A^\top A = I_r$. Let $s_i = \|\Delta B_i\|_{2,1}$ and $s_j = \|\Delta B_j\|_{2,1}$. Then

$$|\langle \Delta W_i, \Delta W_j \rangle_F| \leq s_i s_j. \quad (13)$$

Proof. Starting from the definition,

$$\langle \Delta W_i, \Delta W_j \rangle_F = \text{Tr}((\Delta W_i)^\top \Delta W_j) = \text{Tr}((A\Delta B_i)^\top (A\Delta B_j)). \quad (14)$$

Using $A^\top A = I_r$, this becomes $\text{Tr}(\Delta B_i^\top \Delta B_j)$, which decomposes row-wise:

$$\text{Tr}(\Delta B_i^\top \Delta B_j) = \sum_{l=1}^r \langle (\Delta B_i)_{l,:}, (\Delta B_j)_{l,:} \rangle. \quad (15)$$

Applying the triangle inequality and Cauchy-Schwarz,

$$\left| \sum_{l=1}^r \langle (\Delta B_i)_{l,:}, (\Delta B_j)_{l,:} \rangle \right| \leq \sum_{l=1}^r \|(\Delta B_i)_{l,:}\|_2 \|(\Delta B_j)_{l,:}\|_2. \quad (16)$$

Let $u_l = \|(\Delta B_i)_{l,:}\|_2$ and $v_l = \|(\Delta B_j)_{l,:}\|_2$ (non-negative). Then $\sum_l u_l v_l \leq (\sum_l u_l)(\sum_l v_l) = \|\Delta B_i\|_{2,1} \|\Delta B_j\|_{2,1} = s_i s_j$, proving Eq. 13. $\square$

Interpretation. Eq. 16 shows that interference is driven by shared active rows: the product $\|(\Delta B_i)_{l,:}\|_2 \|(\Delta B_j)_{l,:}\|_2$ is non-zero only when both tasks use rank dimension l . Compared to standard LoRA (where $A^\top A \neq I$ and cross-dimension correlations appear), SO-LoRA isolates geometry in the coefficient space. Minimizing the $\ell_{2,1}$ norm in Eq. 8 therefore directly suppresses overlap and tightens the bound in Eq. 13.

IV. EXPERIMENTS

We evaluate SO-LoRA on standard class-incremental learning (CIL) benchmarks and compare against strong rehearsal-free prompt- and LoRA-based baselines. Across longer task sequences and severe distribution shifts, SO-LoRA achieves higher retention and a steadier learning trajectory, while keeping a constant adapter footprint.

A. Experimental Setup

Datasets. We use four CIL benchmarks with disjoint class splits per task. ImageNet-R [18] is our primary scalability benchmark, evaluated under 5/10/20-task partitions. To test robustness and granularity, we additionally report results on ImageNet-A [19], CIFAR-100 [20], and CUB-200 [21], each configured as a 10-task sequence.

Metrics. Let $a_{t,i}$ be the accuracy on task i after training through task t . We report (i) *Average Accuracy* $\text{Acc} = \frac{1}{T} \sum_{i=1}^T a_{T,i}$, which measures final performance after the full stream, and (ii) *Average Anytime Accuracy* $\text{AAA} =$

$\frac{1}{T} \sum_{t=1}^T \frac{1}{t} \sum_{i=1}^t a_{t,i}$, which summarizes stability at every task boundary.

Baselines. We compare with representative rehearsal-free prompt methods (L2P [9], DualPrompt [10], CODA-Prompt [11], HiDe-Prompt [12]) and PEFT baselines (O-LoRA [14], InfLoRA [15], SD-LoRA [7]).

B. Implementation Details

Training. We follow prior PEFT-CIL settings: a ViT-B/16 [22] backbone pre-trained on ImageNet-21k [23] is frozen, and SO-LoRA is inserted into the query/value projections with rank $r=40$. We minimize cross-entropy plus an $\ell_{2,1}$ penalty ($\lambda=10^{-4}$) using Adam for 15 epochs per task (learning rate 5×10^{-3} , batch size 16). The mask update uses $\alpha=0.8$. We report mean $\pm$ std over five random seeds.

Inference. To reduce classifier bias in incremental evaluation, we use Nearest Class Mean (NCM). After each task, class prototypes are computed as $\mu_c = \frac{1}{|\mathcal{D}_c|} \sum_{x \in \mathcal{D}_c} f_\theta(x)$, and prediction uses cosine similarity $\hat{y} = \arg \max_c \frac{f_\theta(x)^\top \mu_c}{\|f_\theta(x)\| \|\mu_c\|}$.

C. Main Results

Tables I–II summarize the main comparison between comprehensive analysis of SO-LoRA against current state-of-the-art rehearsal-free methods.

TABLE I: Performance on ImageNet-R under different task granularities. Best is bold.

Method	IN-R (5)		IN-R (10)		IN-R (20)	
	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$
L2P	71.54 (0.64)	66.30 (0.35)	70.62 (0.23)	64.59 (0.48)	67.35 (0.53)	59.97 (0.31)
DualPrompt	73.28 (0.41)	71.43 (0.46)	71.23 (0.74)	67.72 (0.54)	68.61 (0.74)	62.88 (0.80)
CODA	78.68 (0.23)	75.29 (0.52)	77.11 (0.71)	73.03 (0.63)	73.71 (0.55)	68.53 (0.61)
HiDe	77.42 (0.21)	72.45 (0.31)	76.84 (0.35)	72.63 (0.61)	75.88 (0.25)	71.56 (0.66)
O-LoRA	76.71 (1.81)	72.76 (1.52)	71.30 (1.02)	64.13 (1.86)	68.04 (0.36)	60.98 (0.98)
InfLoRA	81.63 (0.27)	77.83 (0.33)	80.24 (0.36)	77.04 (0.68)	76.78 (0.86)	70.94 (0.90)
SD-LoRA	82.85 (0.40)	79.02 (0.28)	81.05 (0.42)	77.92 (0.22)	80.62 (0.50)	75.43 (0.57)
SO-LoRA	83.94 (0.24)	79.78 (0.37)	83.71 (0.25)	78.83 (0.21)	83.18 (0.29)	78.15 (0.12)

TABLE II: Performance on ImageNet-A, CIFAR-100, and CUB-200 (10 tasks). Best is bold.

Method	IN-A		C100		CUB	
	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$	AAA $\uparrow$	Acc $\uparrow$
L2P	50.61 (0.21)	41.17 (0.39)	87.58 (0.52)	82.76 (0.98)	69.30 (0.39)	56.48 (0.29)
DualPrompt	54.86 (0.53)	43.07 (0.29)	90.57 (0.23)	83.88 (0.48)	78.57 (0.54)	67.26 (0.49)
CODA	56.30 (0.84)	45.94 (0.49)	91.37 (0.27)	86.82 (0.69)	82.36 (0.40)	72.41 (0.63)
HiDe	55.72 (0.44)	44.67 (0.82)	91.17 (0.34)	86.53 (0.31)	81.71 (0.51)	74.15 (0.72)
O-LoRA	54.17 (0.81)	44.46 (1.48)	89.96 (1.43)	82.51 (1.52)	79.66 (1.57)	70.12 (0.40)
InfLoRA	61.63 (0.19)	51.02 (0.26)	91.44 (0.46)	86.09 (0.28)	81.43 (0.46)	71.13 (0.67)
SD-LoRA	63.24 (0.20)	55.35 (0.47)	91.71 (0.33)	87.28 (0.23)	85.15 (0.55)	77.38 (0.35)
SO-LoRA	67.89 (0.63)	60.27 (0.28)	90.88 (0.17)	86.61 (0.06)	87.71 (0.24)	81.58 (0.43)

Scalability and Long-horizon retention. SO-LoRA ranks first on ImageNet-R for all task granularities. Its degradation from 5 to 20 tasks is small (Acc: from 79.78% to 78.15%), whereas O-LoRA drops sharply (Acc: from 72.76% to 60.98%), consistent with subspace drift over long horizons. In the hardest 20-task setting, SO-LoRA improves over SD-LoRA by +2.72 Acc, suggesting better scalability when a single shared adapter must serve all tasks.

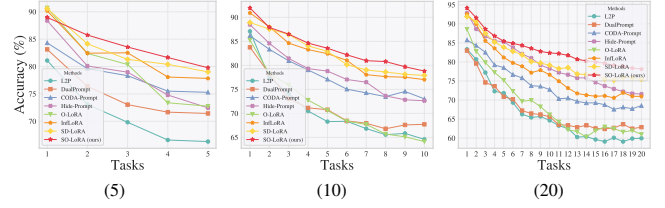


Fig. 2: Average accuracy on ImageNet-R across sequence lengths.

TABLE III: Ablation on 10-task benchmarks. Impact of group sparsity and gradient-aware mask.

Method	ImageNet-A		ImageNet-R	
	AAA	Acc	AAA	Acc
SO-LoRA	67.89 _(0.63)	60.27 _(0.28)	83.71 _(0.25)	78.83 _(0.21)
w/o sparsity	66.86 _(0.96)	59.30 _(0.54)	83.52 _(0.28)	78.77 _(0.31)
w/o mask	66.31 _(1.25)	58.15 _(0.83)	83.40 _(0.24)	78.37 _(0.14)

Robustness under domain shift. ImageNet-A induces a substantial distribution shift. SO-LoRA achieves the best performance (Acc 60.27, AAA 67.89), exceeding SD-LoRA by nearly five Acc points. This gap supports the role of the gradient-aware mask: when current data depart from the pre-training manifold, explicitly protecting historically important directions becomes critical and prompt-only strategies become less reliable.

Fine-grained transfer. On CUB-200, SO-LoRA attains the highest accuracy (81.58), indicating that sparse recombinations over an orthogonal basis can capture subtle cues without amplifying interference. On CIFAR-100, SO-LoRA remains competitive and close to the best baseline, suggesting that the proposed constraints preserve plasticity.

D. Ablation Study

We ablate group sparsity and the gradient-aware mask to isolate their effects on retention (Acc) and stability (AAA).

Effect of group sparsity. On ImageNet-R, removing $\ell_{2,1}$ only marginally changes performance, suggesting that the fixed orthogonal basis already provides strong geometric separation for in-domain tasks. Under the larger shift of ImageNet-A, sparsity becomes more beneficial (Acc: from 60.27% to 59.30%), consistent with row-sparse task codes reducing overlap when tasks are less aligned.

Effect of gradient-aware masking. Masking yields the most consistent gains. Removing it causes the largest drop on ImageNet-A (Acc: from 60.27% to 58.15%) and a clearer penalty on ImageNet-R with 20 tasks (Acc: from 78.15% to 77.30%), indicating that long horizons still benefit from explicitly protecting historically important directions even within an orthogonal parameterization.

E. Discussion

Rank Sensitivity and Basis Capacity. We vary the subspace rank $r \in [10, 90]$ (Fig. 3) to quantify the capacity–interference trade-off. Accuracy rises quickly for small ranks, suggesting

TABLE IV: Ablation across sequence lengths on ImageNet-R.

Method	5 tasks		20 tasks	
	AAA	Acc	AAA	Acc
SO-LoRA	83.94 _(0.24)	79.78 _(0.37)	83.18 _(0.29)	78.15 _(0.12)
w/o sparsity	83.11 _(0.59)	79.71 _(0.38)	83.20 _(0.33)	78.07 _(0.14)
w/o mask	83.41 _(0.32)	79.03 _(0.34)	82.66 _(0.31)	77.30 _(0.29)

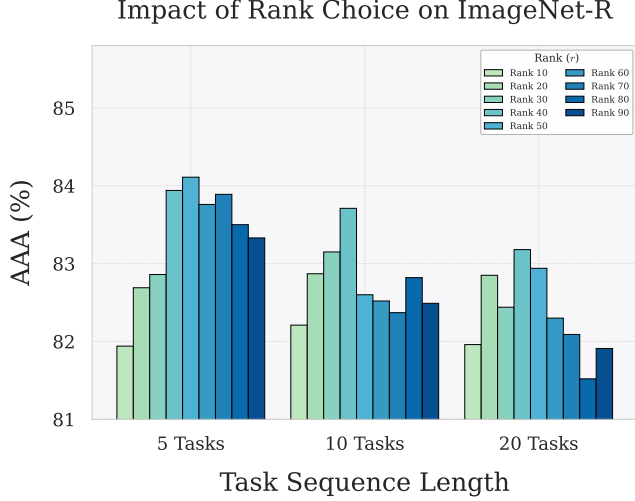


Fig. 3: **Rank sensitivity.** Performance peaks at $r = 40$, balancing capacity and interference.

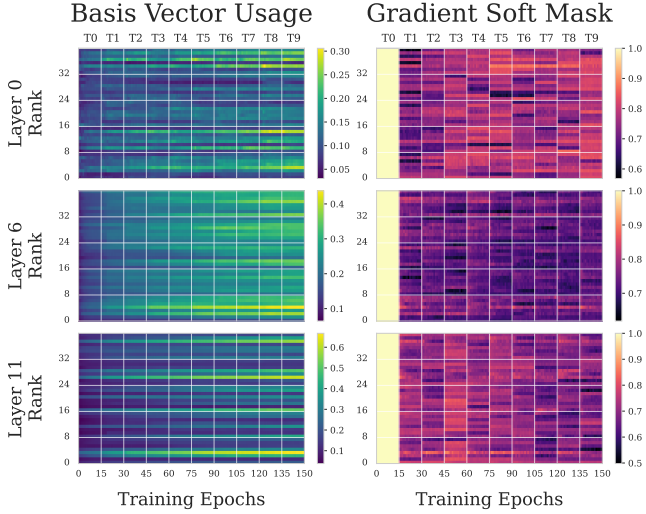


Fig. 4: **Internal dynamics.** Basis usage (Left) and gradient masks (Right) show dynamic allocation.

that a minimal orthogonal “vocabulary” is needed to cover ImageNet-R variation. Gains saturate around $r = 40$, and performance declines once $r \gtrsim 50$. Larger bases increase degrees of freedom and weaken the practical effect of sparsity, allowing task solutions to spread across many directions and increasing overlap. We therefore set $r = 40$ as a stable operating point: enough capacity to model new tasks without inviting unnecessary interference.

Basis Vector Usage and Subspace Evolution. Figure 4

(Left) visualizes the evolution of $B^{(t)}$ via row-norm heatmaps. The clear banded structure indicates that each task activates only a small subset of basis directions, consistent with the intended group sparsity behavior. Notably, inactive directions early in training become selectively recruited later, implying a “reserve” of capacity that is gradually allocated as new concepts arrive. This staggered recruitment helps avoid early saturation of the fixed basis and supports longer task horizons. **Basis Vector Usage and Subspace Evolution.** Figure 4 (Right) shows how the gradient-aware mask distributes stability and plasticity across layers. Early layers exhibit sharper separation between frozen and adaptable dimensions, consistent with protecting general low-level primitives (edges/textures) once they stabilize ($S_j \approx 0$ for high-importance directions). Deeper layers display smoother, less binary masks, reflecting the need for controlled adaptation of higher-level semantics that may partially overlap across tasks. Overall, the mask behaves as intended: it strongly protects historically sensitive directions while leaving sufficient flexibility where transfer and re-organization are beneficial.

V. CONCLUSION

In this paper, we proposed SO-LoRA, a parameter-efficient continual learning framework that avoids the usual choice between shared-adapter interference and per-task memory growth. By representing updates as sparse compositions over a fixed orthogonal basis and modulating learning with a gradient-aware soft mask, SO-LoRA limits destructive overlap while keeping constant adapter size. Across benchmarks, it delivers strong retention in long task streams and remains competitive under substantial domain shift, offering a practical path toward lifelong adaptation of large foundation models.

Algorithm 1 SO-LoRA Training for a New Task $\mathcal{T}_t$

Input: Pre-trained weights W_0 , frozen basis A , shared matrix $B_{\text{shared},t-1}$, mask M_{t-1} , task data $\mathcal{D}_t$.

Output: Updated $B_{\text{shared},t}$, updated mask M_t , class prototypes P_t .

```
1:  $B_{\text{shared}} \leftarrow B_{\text{shared},t-1}$ 
2: Extend classifier head  $h_\phi$  for new classes in  $C_t$ 
3: Freeze  $W_0$ ,  $A$ , and past classifier weights
4: for each epoch do
5:   for each batch  $(x, y) \in \mathcal{D}_t$  do
6:     ▷ Forward
7:      $W_{\text{adapted}} \leftarrow W_0 + AB_{\text{shared}}$ 
8:      $f(x) \leftarrow \text{Backbone}(x; W_{\text{adapted}})$ 
9:      $\hat{y} \leftarrow h_\phi(f(x))$ 
10:    ▷ Gradients
11:     $\mathcal{L}_{\text{task}} \leftarrow \mathcal{L}_{\text{CE}}(\hat{y}, y)$ 
12:     $g_{\text{task}} \leftarrow \nabla_{B_{\text{shared}}} \mathcal{L}_{\text{task}}$ 
13:     $g_{\text{reg}} \leftarrow \nabla_{B_{\text{shared}}} (\lambda \|B_{\text{shared}}\|_{2,1})$ 
14:    ▷ Gradient-aware modulation
15:    for  $i \in \{1, \dots, r\}$  do
16:       $\phi_i \leftarrow \frac{\|(g_{\text{task}})_{i,:}\|_2}{\max_j \|(g_{\text{task}})_{j,:}\|_2 + \epsilon}$ 
17:       $S_i \leftarrow 1 - (M_{t-1,i}(1 - \phi_i))$ 
18:       $(g'_{\text{task}})_{i,:} \leftarrow S_i \cdot (g_{\text{task}})_{i,:}$ 
19:    end for
20:     $g' \leftarrow g'_{\text{task}} + g_{\text{reg}}$ 
21:    Update  $B_{\text{shared}}$  and new classifier weights using  $g'$ 
22:  end for
23: end for
24:  $B_{\text{shared},t} \leftarrow B_{\text{shared}}$ 
25: ▷ Prototypes (NCM)
26: For each class  $c \in C_t$ :  $\mu_c \leftarrow \frac{1}{|\mathcal{D}_c|} \sum_{(x_i, y_i) \in \mathcal{D}_c} \text{Backbone}(x_i; W_0 + AB_{\text{shared},t})$ 
27:  $P_t \leftarrow \{\mu_c\}_{c \in C_t}$ 
28: ▷ Mask update (EMA of row-gradient norms)
29: Compute  $g_{\text{final}} \leftarrow \nabla_{B_{\text{shared}}} \mathcal{L}_{\text{CE}}$  over  $\mathcal{D}_t$  using  $B_{\text{shared},t}$ 
30: For  $i \in \{1, \dots, r\}$ :  $\omega_{t,i} \leftarrow \|(g_{\text{final}})_{i,:}\|_2$ 
31:  $M_t \leftarrow \alpha \cdot \frac{\omega_t}{\max_j (\omega_{t,j}) + \epsilon} + (1 - \alpha) \cdot M_{t-1}$ 
```

- [8] X. Liu and X. Chang, “LoRA subtraction for drift-resistant space in exemplar-free continual learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025.
- [9] Z. Wang et al., “Learning to prompt for continual learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 139–149.
- [10] Z. Wang et al., “Dualprompt: Complementary prompting for rehearsal-free continual learning,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2022, pp. 631–648.
- [11] J. S. Smith et al., “Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 11909–11919.
- [12] L. Wang, J. Xie, X. Zhang, M. Huang, H. Su, and J. Zhu, “Hierarchical decomposition of prompt-based continual learning,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2023, vol. 36, pp. 60527–60544.
- [13] G. Zhang, L. Wang, G. Kang, L. Chen, and Y. Wei, “Slca: Slow learner with classifier alignment for continual learning on a pre-trained model,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 19091–19101.
- [14] X. Wang et al., “Orthogonal subspace learning for language model continual learning,” in *Findings of the Assoc. Comput. Linguist. (EMNLP)*, 2023, pp. 10658–10671.
- [15] Y. S. Liang and W. J. Li, “InfloRA: Interference-free low-rank adaptation for continual learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024, pp. 23638–23647.
- [16] J. Pfeiffer et al., “Adapterfusion: Non-destructive task composition for transfer learning,” in *Proc. Eur. Chapter Assoc. Comput. Linguist. (EACL)*, 2021, pp. 487–503.
- [17] C. S. Huang et al., “LoRAhub: Efficient cross-task generalization via dynamic LoRA composition,” in *Conf. Learn. Mach. (COLM)*, 2024.
- [18] D. Hendrycks et al., “The many faces of robustness: A critical analysis of out-of-distribution generalization,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 8340–8349.
- [19] D. Hendrycks et al., “Natural adversarial examples,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 15262–15271.
- [20] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Tech. Rep., University of Toronto, 2009.
- [21] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The Caltech-UCSD birds-200-2011 dataset,” Tech. Rep. CNS-TR-2011-001, California Institute of Technology, 2011.
- [22] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in *Int. Conf. Learn. Represent. (ICLR)*, 2021.
- [23] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2009, pp. 248–255.

REFERENCES

- [1] D. W. Zhou, H. L. Sun, J. Ning, H. J. Ye, and D. C. Zhan, “Continual learning with pre-trained models: A survey,” in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2024, pp. 8363–8371.
- [2] D. W. Zhou, Q. W. Wang, Z. H. Qi, H. J. Ye, D. C. Zhan, and Z. Liu, “Deep class-incremental learning: A survey,” *arXiv preprint arXiv:2302.03648*, 2023.
- [3] E. J. Hu et al., “LoRA: Low-rank adaptation of large language models,” in *Int. Conf. Learn. Represent. (ICLR)*, 2022.
- [4] M. Wistuba, P. T. Sivaprasad, L. Balles, and G. Zappella, “Continual learning with low-rank adaptation,” in *NeurIPS Workshop on Distribution Shifts*, 2023.
- [5] J. S. Smith et al., “Continual diffusion: Continual customization of text-to-image diffusion with C-LoRA,” *Trans. Mach. Learn. Res. (TMLR)*, 2024.
- [6] A. Chavan et al., “One-for-all: Generalized LoRA for parameter-efficient fine-tuning,” *arXiv preprint arXiv:2306.07967*, 2023.
- [7] Y. Wu et al., “Sd-loRA: Scalable decoupled low-rank adaptation for class-incremental learning,” in *Int. Conf. Learn. Represent. (ICLR)*, 2025.