ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

Multi-Layer Distillation and Prototype Replay for Class-Incremental Learning

Ce Zhao[1,a], Yiwen Zhang[1,b], Lifeng Yao[1,c], Jingya Wang[1,d,*], Yonghao Chen[1,e], Naifu Ye[2,f]

[1]College of Information and Cyber Security, People's Public Security University of China, Beijing, China
[2]School of Policing Information, Shandong Police College, Jinan, China
[a]2021211460@stu.ppsuc.edu.cn, [b]2022211475@stu.ppsuc.edu.cn, [c]2022211471@stu.ppsuc.edu.cn, [d]wangjingya@ppsuc.edu.cn, [e]2022211456@stu.ppsuc.edu.cn, [f]1025441698@qq.com
**Corresponding author*

Abstract: *The primary challenge in class-incremental learning is mitigating catastrophic forgetting, where a model's performance on previously learned tasks deteriorates drastically as it adapts to new tasks. This problem becomes even more pronounced in real-world applications, where the new data is constantly emerging, and retraining from scratch is not feasible. In this paper, we propose a class-incremental learning method called MDPR, which integrates several techniques, including self-supervised data augmentation, multi-layer knowledge distillation, and enhanced prototype replay. These strategies collectively improve the feature diversity extracted by the model, preserve knowledge of previous tasks, and address the issue of class imbalance in the classifier. Experimental results on the CIFAR-100 benchmarks demonstrate that MDPR effectively mitigates catastrophic forgetting, achieving high performance in class-incremental learning tasks.*

Keywords: *Class-Incremental Learning, Catastrophic Forgetting, Lifelong learning, Image Classification*

1. Introduction

With the rapid development of information technology and data acquisition techniques, the volume of data in modern society is growing exponentially. Traditional machine learning methods, which typically rely on large, static datasets for offline training, often incur high computational and storage costs. This is especially problematic when dealing with massive, constantly changing data, as these methods lack the flexibility to adapt to dynamic environments. As a result, the design of machine learning algorithms capable of continuous learning and adaptation to new data has become a crucial area of research.

Incremental learning offers a promising solution by allowing models to progressively adapt to new data without the need to retrain from scratch [1][2]. Unlike traditional batch learning, incremental learning reduces computational and storage demands while improving the model's ability to handle evolving data. It has shown great promise in applications such as sensor data streams, social media content analysis, and financial market forecasting [3]. However, incremental learning also faces significant challenges, including mitigating catastrophic forgetting [4][5], balancing stability with adaptability to new data [6], and optimizing algorithm efficiency under constraints of limited computational and storage resources [7]. Despite these challenges, incremental learning has demonstrated remarkable potential in fields such as online learning systems (e.g., recommendation systems and ad click prediction), computer vision (e.g., object detection and facial recognition), and natural language processing (e.g., text classification and sentiment analysis) [8][9].

2. Related Work

Incremental learning has advanced significantly in addressing catastrophic forgetting. Methods can be generally categorized into three types: replay-based, regularization-based, and dynamic architecture-based methods.

Replay-based methods attempt to replay samples from previous tasks to prevent forgetting while

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

learning new tasks. These methods can be further divided into rehearsal, pseudo rehearsal, and model update constraint. Rehearsal methods, like iCaRL [5], combine old and new task data for training, with keeping exemplars kept in the memory. These methods have been extended with better sample selecting and managing techniques [10][11][12]. Pseudo Rehearsal generates pseudo-samples or stores intermediate representations of previous tasks. Shin et al. [13] introduced Deep Generative Replay (DGR), using GANs to generate pseudo-data. Constrained Update limits parameter updates using stored old samples, as seen in GEM [8] and its extension A-GEM [14]. GSS [15] adapts this approach to data-incremental learning. While effective, replay-based methods require significant storage and computational resources, raising concerns about data privacy.

Regularization-based methods mitigate catastrophic forgetting by adding regularization terms to the loss function, restricting parameter updates without relying on replay. These can be divided into model-performance-based and model-parameter-based regularization. Model-performance-based methods, like LwF [2], use knowledge distillation to preserve old task knowledge while training new tasks. Other approaches, like AFC [16] and FOSTER [17], further optimize feature consolidation and compression. Model-parameter-based methods, such as EWC [4] and SI [6], limit parameter changes by measuring parameter importance, preventing catastrophic forgetting while training on new tasks. These methods allow knowledge sharing between tasks without the need for stored data, but generally perform less effectively than replay-based methods.

Dynamic architecture-based methods add new units to the model to avoid forgetting. PNN [3] introduces new units on top of the old model, while PackNet [18] prunes redundant parameters to make room for new tasks. Piggyback [19] uses masks to activate relevant parameters in a pre-trained network. These methods effectively prevent forgetting by fixing old parameters, but often require task labels during testing, making them suitable primarily for task-incremental learning.

3. Methodology

3.1. Preliminary

Class-Incremental Learning (CIL) is a specific type of incremental learning that focuses on gradually introducing new classes, where each new task involves learning a set of new categories [7]. The objective of class-incremental learning is to learn a sequence of tasks, enabling the model to classify test samples of any learned classes. For the i-th task T_i in the n-task sequence $\left(T_1, T_2, ..., T_{n-1}, T_n\right)$, the corresponding dataset is denoted as D_i, which includes both the training set $D_{i,train}$ and the test set $D_{i,test}$. The training set $D_{i,train}$ contains $m_{i,train}$ samples. The j-th sample in $D_{i,train}$ is denoted as $\left(x_{i,train,j}, y_{i,train,j}\right)$, which is an image-label pair. The set of all image samples in it is denoted as $X_{i,train}$, with the set of all labels in it being $Y_{i,train}$. The test sets follow the same notation as the training sets. Typically, the following conditions hold:

- For any task, the class labels in both the training and test sets are identical, i.e., $Y_{i,train} = Y_{i,test}$, while $1 \le i \le N$.
- The classes in the training sets of any two distinct tasks have no overlap, i.e., $Y_{i,train} \cap Y_{j,train} = \varnothing$, while $1 \le i < j \le n$.

When learning task T_i, the goal is to minimize the empirical loss on the set $\bigcup_{j=1}^{i} D_{i,test}$, which includes all samples from every test set so far. In this paper, we follow the strict class-incremental learning setup, where no raw or generated samples from previous tasks are allowed during the training of new tasks [20].

3.2. Overview of MDPR

Our proposed class-incremental learning method, named MDPR, involves multi-layer knowledge

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

distillation (MLKD) to retain previous task knowledge and self-supervised learning (SSL) to enrich feature extraction. Additionally, inspired by replay-based methods, MDPR stores class prototypes for enhanced replay in the classifier, addressing the class imbalance issue. Figure 1 illustrates the overall framework.

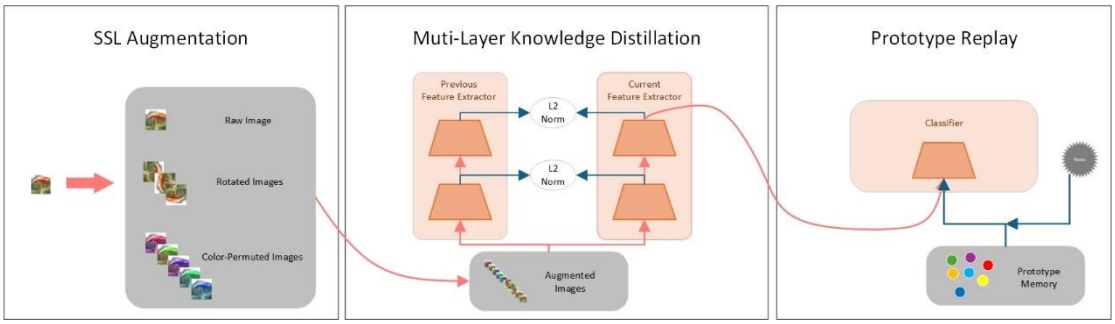

Figure 1: Framework of MDPR.

The classification model used in this paper can be denoted as:

$$Model\left(x,\phi,\theta\right)=G\left(F\left(x,\theta\right),\phi\right), \tag{1}$$

where $F\left(x,\theta\right)$ is the l-layer feature extractor and $G\left(feature,\phi\right)$ is the classifier with

$$feature=F\left(x,\theta\right)=f^{l}\left(...f^{2}\left(f^{1}\left(x,\theta^{1}\right),\theta^{2}\right)...,\theta^{l}\right), \tag{2}$$

$f^{j}\left(x,\theta^{j}\right)$ is the j-th layer of the feature extractor.

3.3. SSL Data Augmentation

In incremental learning, training the feature extractor with more data enhances feature diversity and model generalization. Self-supervised learning (SSL) augmentation further boosts feature robustness. Inspired by PASS, we propose an improved augmentation strategy combining Color Channel Perturbation (CCP) [21] and rotation augmentation [22], leveraging spatial and channel-level transformations to enrich representations and reduce classifier bias.

For any sample $\left(x_{i,train,j},y_{i,train,j}\right)$ in the $D_{i,train}$ from T_i, we apply the following augmentations during training:

- Spatial Transformation (Rotation): The image is randomly rotated by 90°, 180°, or 270°, generating an augmented image $rotate\left(x_{i,train,j}\right)$ with a corresponding label. This introduces geometric variations, encouraging the model to stay sensitive to spatial features.
- Channel Transformation (Color Perturbation): The RGB channels of the image are randomly permuted, generating an augmented image $permute\left(x_{i,train,j}\right)$ with a corresponding label. This enhances the model's robustness to color changes and helps it learn more texture and structural details.

As for the original sample $\left(x_{i,train,j},y_{i,train,j}\right)$, it is also assigned a new label for the sake of consistency. It is worth mentioning that the classifier will discard enhanced outputs during inference.

3.4. Multi-Layer Distillation

Knowledge distillation is a key technique for model compression and knowledge transfer, alleviating catastrophic forgetting by linking old and new tasks. In incremental learning, the teacher model's soft labels help the student model retain knowledge of previous tasks while learning new ones. Multi-layer distillation (MLKD), which aligns intermediate layer features, extends this approach by preserving the model's representational capacity, making it particularly effective for complex tasks. Our method combines multi-layer distillation with self-supervised data augmentation, significantly reducing

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

catastrophic forgetting by retaining the feature extractor's knowledge from previous tasks.

To ensure the feature extractor retains knowledge of past tasks, we apply an L2 regularization to align the intermediate and final features extracted by the current model with those by the model right after the previous task:

$$L_{dist} = \left\| feature_{current}^{middle} - feature_{prev}^{middle} \right\|_2 + \left\| feature_{current}^{l} - feature_{prev}^{l} \right\|_2, \tag{3}$$

where

$$feature^{i} = f^{i}\left(...f^{2}\left(f^{1}\left(x,\theta^{1}\right),\theta^{2}\right)...,\theta^{i}\right), \tag{4}$$

In the case of the ResNet-18 feature extractor used in the experiments, we perform distillation on the intermediate down sampling layer (with output size of 128x16x16) and the final feature layer (output size 512).

3.5. Prototype Replay

In class-incremental learning, class imbalance often leads to bias toward the current task's classes [23]. This problem arises because, during training, the model can only access data from the current task, even though it retains knowledge of previous tasks. To address this, we adopt the enhanced prototype replay (PR) from [24]. By storing prototypes for each class of previous tasks and applying Gaussian noise perturbations to these prototypes during subsequent tasks, we mitigate feature space distortion and reduce classifier bias.

We compute the prototype of each class as the mean of all its training samples in the feature space after the task training. To compute the prototype of the class with label k in $D_{i,train}$:

$$proto_{i,j} = \frac{1}{s_{i,train}^{j}} \times \sum_{x \in X_{i,train}^{j}} F_i\left(x,\theta_i\right), \tag{5}$$

where $x_{i,train}^{j}$ is the set of all samples with label j in $D_{i,train}$ and $s_{i,train}^{j}$ is its number. These prototypes are stored and then used in later tasks. When replayed, the old prototypes are enhanced with Gaussian noise:

$$aug\left(proto_{i,j}\right) = proto_{i,j} + e \times r, \tag{6}$$

where $e \sim N\left(0,1\right)$ and r is the scaling factor, dynamically updated based on the prototype variance:

$$r_i^2 = \frac{1}{c_i} \times \left(c_{i-1} \times r_{i-1}^2 + \sum_{j \in Y_{i,train}} \frac{Tr\left(\Sigma_{i,j}\right)}{d} \right), \tag{7}$$

Where c_i is the total number of the ever seen class labels until T_i, $Tr\left(\Sigma_{i,j}\right)$ is the covariance matrix for the features of class j in $D_{i,train}$. The prototype classification loss is computed as:

$$L_{proto} = \sum_{j=1}^{i-1} \sum_{k \in Y_{l,train}} L_{CE}\left(G\left(aug\left(proto_{j,k}\right)\right), y_{j,k}^{proto}\right), \tag{8}$$

where L_{CE} prototype is the Cross Entropy function, $y_{j,k}^{proto}$ is the corresponding class label of $proto_{j,k}$. This approach ensures that the classifier retains knowledge of previous task prototypes, preventing bias accumulation and mitigating class imbalance.

The final loss function combines classification loss, knowledge distillation loss, and prototype replay

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

loss:

$$L_{total} = L_{clf} + \lambda_{dist} \times L_{dist} + \lambda_{proto} \times L_{proto}, \tag{9}$$

where

$$L_{clf} = \sum_{(x,y)\in D_{i,train}} L_{CE}\left(Model\left(x,\phi,\theta\right),y\right), \tag{10}$$

with two weight hyperparameters λ_{dist} and λ_{proto}. We set $\lambda_{dist} = 9$ and $\lambda_{proto} = 10$ in our experiments.

4. Experiments

4.1. Setup

We conducted experiments on the CIFAR-100 dataset, which contains 100 classes, each with 600 color images, totaling 60,000 samples (50,000 for training, 10,000 for testing). The images have a resolution of 32×32 pixels in RGB format. For class-incremental learning, we follow the setup from [24], including two configurations: 1) the first task contains 50 classes, and the remaining 5 tasks each contain 10 classes; 2) the first task has 50 classes, and the remaining 10 tasks each have 5 classes.

We evaluated the model using three metrics: Final Accuracy (accuracy on the joint test set after all tasks), Average Incremental Accuracy (average value of Final Accuracies on all tasks), and Average Forgetting Rate (average accuracy drop of previous tasks and after all tasks). All metrics are widely adopted in class-incremental learning researches [24] [25] [26].

We used the ResNet-18 backbone for feature extraction. The input image (32×32×3) undergoes a 7×7 convolution and max-pooling, reducing the feature map size from 224×224 to 56×56. The model then performs residual convolution across four stages, progressively down sampling and increasing the number of channels. After the final stage, global average pooling compresses the feature map to a 512-length vector.

The model is trained for 100 epochs per task with the batch size of 64, using the Adam optimizer [27], initial learning rate of 0.001, and weight decay of 0.0004. The learning rate decays by 0.1 after the 45th and 90th epochs.

4.2. Results

We evaluated our method against several morden exemplar-free methods, including EWC [4], LwF [2], LwM [28], MUC [29], SSRE [30], and PASS [24]. The results, shown in Table 1 and Table 2, highlight the significant advantages of MDPR over other methods.

Table 1: Accuracy Metrics.

	5 steps		10 steps	
Method	Final Accuracy	Average Incremental Accuracy	Final Accuracy	Average Incremental Accuracy
EWC	10.32	11.86	8.56	18.16
LwF	15.34	18.32	7.19	19.76
LwM	38.23	40.11	20.90	23.31
MUC	39.15	40.31	19.94	22.59
SSRE	43.92	51.78	43.67	49.40
PASS	54.65	61.32	45.76	59.63
MDPR (Ours)	**58.00**	**67.42**	**48.96**	**62.81**

On the CIFAR-100 dataset, MDPR consistently outperforms the other methods across both final accuracy and average incremental accuracy for both 5-step and 10-step incremental learning tasks. In the 5-step setup, MDPR achieves a final accuracy of 58.00%, significantly outperforming PASS (54.65%), the best among the other methods. Additionally, MDPR leads in average incremental accuracy with 67.42%, compared to PASS's 61.32%. In the 10-step setup, MDPR maintains its superior performance,

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

reaching a final accuracy of 48.96% and an average incremental accuracy of 62.81%, again surpassing PASS, which achieves 45.76% and 59.63%, respectively. In contrast, methods like EWC, LwF, and LwM exhibit significantly lower performance, with final accuracies ranging from 8.56% to 38.23% for 10 steps, underscoring the effectiveness of MDPR in retaining knowledge across multiple tasks.

Table 2: Average Forgetting Rate Metrics (Lower is Better).

Method	5 steps	10 steps
SSRE	30.05	35.78
PASS	25.20	**30.25**
MDPR (Ours)	**21.50**	34.48

To assess catastrophic forgetting, we present the average forgetting rates for MDPR, PASS, and SSRE in Table 2. In the 5-step setup, MDPR achieves the lowest forgetting rate of 21.50%, outperforming PASS (25.20%) and SSRE (30.05%). In the 10-step setup, although MDPR's forgetting rate increases slightly to 34.48%, it remains the low compared to PASS (30.25%) and SSRE (37.78%).

The above results demonstrate that MDPR not only delivers superior performance in accuracy but also effectively mitigates forgetting, outperforming PASS and SSRE in retaining knowledge across tasks without the need for exemplar storage.

4.3. Ablation Study

Our proposed method consists of three key components: SSL augmentation, multi-layer knowledge distillation for the feature extractor, and enhanced prototype replay for the classifier. We analyze the impact of each component on the final results, as shown in the ablation study in Table 3.

Table 3: 5-Step Benchmark Ablation Study.

Method	Final Accuracy
MLKD	8.22
MLKD+SSL	8.49
MLKD+PR	51.65
MLKD+SSL+PR	58.00

From the results, we observe that using only knowledge distillation or self-supervised data augmentation leads to very low accuracy due to the classifier's inability to retain old knowledge, resulting in severe class imbalance. Prototype replay effectively mitigates this issue, and when combined with knowledge distillation, it achieves a significant improvement, boosting accuracy by 43.43%. Furthermore, self-supervised data augmentation further enhances the final classification accuracy.

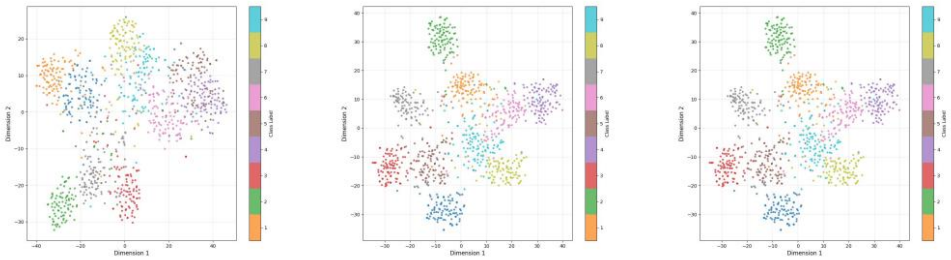

Figure 2: t-SNE Visualization of Finetuning (left), PASS (middle) and MDPR (right).

We also conducted a t-SNE visualization to assess the role of self-supervised data augmentation in enhancing the feature extractor. Figure 2 shows the feature extractor's output after training on the first task. Classes 0, 1, 2, 3, and 4 belong to the first task, while classes 5, 6, 7, 8, and 9 belong to subsequent tasks. From the feature space distribution, we can see that self-supervised data augmentation—using both rotation and channel perturbation—results in more tightly clustered features compared to only applying rotation, as done in [24]. Additionally, the decision boundaries between known and unknown classes are clearer, significantly enhancing the effectiveness of the features extracted by the model

5. Conclusions

This paper presents a simple yet effective class-incremental learning method, MDPR. The key

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

advantage of MDPR lies in its innovative combination of self-supervised data augmentation, multi-layer knowledge distillation, and enhanced prototype replay. This approach effectively improves the feature richness of the extractor, preserves old knowledge during training on subsequent tasks, and mitigates class imbalance issues. Experiments show that MDPR significantly outperforms exemplar-free methods under various settings.

Acknowledgements

This work was partly supported by the Fundamental Research Funds for the Central Universities (Grant Number: 2024JKF13).

References

[1] Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. Neural networks, 113, 54-71.
[2] Li, Z., & Hoiem, D. (2017). Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence, 40(12), 2935-2947.
[3] Rusu, A., Rabinowitz, N., Desjardins, G., et al. (2016). Progressive neural networks. arXiv. https://doi.org/10.48550/arXiv.1606.04671.
[4]Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 114(13), 3521-3526.
[5] Rebuffi, S. A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2001). Incremental classifier and representation learning. In Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5533-5542).
[6] Zenke F, Poole B, Ganguli S. Continual Learning through Synaptic Intelligence. Proceedings of the 34th International Conference on Machine Learning. 2017;70:3987-3995.
[7] Van de Ven, G. M., & Tolias, A. S. (2019). Three scenarios for continual learning. arXiv preprint arXiv:1904.07734.
[8] Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. Advances in neural information processing systems, 30.
[9] Farquhar, S., & Gal, Y. (2018). Towards robust evaluations of continual learning. arXiv preprint arXiv:1805.09733.
[10] Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., & Wayne, G. (2019). Experience replay for continual learning. Advances in neural information processing systems, 32.
[11] 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.
[12] Bang, J., Kim, H., Yoo, Y., Ha, J. W., & Choi, J. (2021). 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).
[13] Shin, H., Lee, J. K., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. Advances in neural information processing systems, 30.
[14] Chaudhry, A., Ranzato, M. A., Rohrbach, M., & Elhoseiny, M. (2018). Efficient lifelong learning with a-gem. arXiv preprint arXiv:1812.00420.
[15] Aljundi, R., Lin, M., Goujaud, B., & Bengio, Y. (2019). Online continual learning with no task boundaries. arXiv preprint arXiv:1903.08671, 3, 2.
[16] Kang, M., Park, J., & Han, B. (2022). Class-incremental learning by knowledge distillation with adaptive feature consolidation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 16071-16080).
[17] Wang, F. Y., Zhou, D. W., Ye, H. J., & Zhan, D. C. (2022, October). Foster: Feature boosting and compression for class-incremental learning. In European conference on computer vision (pp. 398-414). Cham: Springer Nature Switzerland.
[18]Mallya A, Lazebnik S. PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2018 Jun; Salt Lake City, UT. IEEE; 2018:7765–7773.
[19] Mallya, A., Davis, D., & Lazebnik, S. (2018). Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In Proceedings of the European conference on computer vision (ECCV) (pp. 67-82).

ISSN 2616-5775 Vol. 8, Issue 2: 56-63, DOI: 10.25236/AJCIS.2025.080208

[20] Van de Ven, G. M., Tuytelaars, T., & Tolias, A. S. (2022). Three types of incremental learning. Nature Machine Intelligence, 4(12), 1185-1197.
[21] Kantipudi, J., Dubey, S. R., & Chakraborty, S. (2020). Color channel perturbation attacks for fooling convolutional neural networks and a defense against such attacks. IEEE Transactions on Artificial Intelligence, 1(2), 181-191.
[22] Lee, H., Hwang, S. J., & Shin, J. (2020, November). Self-supervised label augmentation via input transformations. In International conference on machine learning (pp. 5714-5724). PMLR.
[23]Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., & Fu, Y. (2019). Large scale incremental learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 374-382).
[24] Zhu, F., Zhang, X. Y., Wang, C., Yin, F., & Liu, C. L. (2021). Prototype augmentation and self-supervision for incremental learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5871-5880).
[25] Yu L, Twardowski B, Liu X, et al. Semantic Drift Compensation for Class-Incremental Learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020:10016-10025. doi:10.1109/CVPR42600.2020.01002.
[26] Goswami, D., Soutif-Cormerais, A., Liu, Y., Kamath, S., Twardowski, B., & van de Weijer, J. (2024). Resurrecting Old Classes with New Data for Exemplar-Free Continual Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 28525-28534).
[27] Kingma, D. P. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[28 Dhar, P., Singh, R. V., Peng, K. C., Wu, Z., & Chellappa, R. (2019). Learning without memorizing. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 5138-5146).
[29] Liu, Y., Parisot, S., Slabaugh, G., Jia, X., Leonardis, A., & Tuytelaars, T. (2020). More classifiers, less forgetting: A generic multi-classifier paradigm for incremental learning. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16 (pp. 699-716). Springer International Publishing.
[30] Zhu, K., Zhai, W., Cao, Y., Luo, J., & Zha, Z. J. (2022). Self-sustaining representation expansion for non-exemplar class-incremental learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 9296-9305).