M2SD:Multiple Mixing Self-Distillation for Few-Shot Class-Incremental Learning

Jinhao Lin[1, 2], **Ziheng Wu**[2], **Weifeng Lin**[1, 2], **Jun Huang**[2], **RongHua Luo**[1*]

[1] South China University of Technology
[2] Alibaba Group
csljh_jasper@mail.scut.edu.cn, zhoulou.wzh@alibaba-inc.com, eelinweifeng@mail.scut.edu.cn,
huangjun.hj@alibaba-inc.com, rhluo@scut.edu.cn

Abstract

Few-shot Class-incremental learning (FSCIL) is a challenging task in machine learning that aims to recognize new classes from a limited number of instances while preserving the ability to classify previously learned classes without retraining the entire model. This presents challenges in updating the model with new classes using limited training data, particularly in balancing acquiring new knowledge while retaining the old. We propose a novel method named **M**ultiple **M**xing **S**elf-**D**istillation (M2SD) during the training phase to address these issues. Specifically, we propose a dual-branch structure that facilitates the expansion of the entire feature space to accommodate new classes. Furthermore, we introduce a feature enhancement component that can pass additional enhanced information back to the base network by self-distillation, resulting in improved classification performance upon adding new classes. After training, we discard both structures, leaving only the primary network to classify new class instances. Extensive experiments demonstrate that our approach achieves superior performance over previous state-of-the-art methods.

Introduction

Deep learning has had a significant impact on computer vision, particularly in image classification (Krizhevsky, Sutskever, and Hinton 2017; Simonyan and Zisserman 2014; Ren et al. 2015). Currently, most computer vision scenarios, including classification, rely on data-driven discriminative tasks, requiring obtaining the specific task data distribution and designing targeted loss functions before model training to achieve better results. Even though numerous publicly available datasets have produced excellent results in various scenarios, the classification task suffers from a significant limitation: it often performs poorly when new classes are added, particularly when the amount of data in these new classes is insufficient and uneven (Ren et al. 2019; Zhu et al. 2021b).

In many real-world scenarios, the situation may worsen when new classes have only a few instances. To address this challenge, researchers have introduced Few-shot Learning(FSL)(Chen et al. 2019) to Class Incremental Learning(CIL)(Wu et al. 2019) and proposed Few-shot Class Incremental Learning (FSCIL)(Ayub and Wagner 2020). One of the significant challenges FSCIL faces is catastrophic forgetting. Introducing new classes to predefined models for a classification task by fine-tuning can potentially lead to an overemphasis on the new classes, possibly resulting in a decline in performance for the existing classes(Kirkpatrick et al. 2017).

To address this problem, various methods(Shi et al. 2021) incorporate more robust regularization with the objective function during incremental sessions, resulting in a minor change in the parameters of the new model relative to the original one. Unlike the regularization methods, the idea behind FACT(Zhou et al. 2022) is to prepare for the arrival of new data in advance, which is akin to forward compatibility in software updates. It is more concerned with obtaining a feature space in the base session of FSCIL that is suitable for subsequent incremental learning sessions.

We are inspired by the idea of FACT and propose a

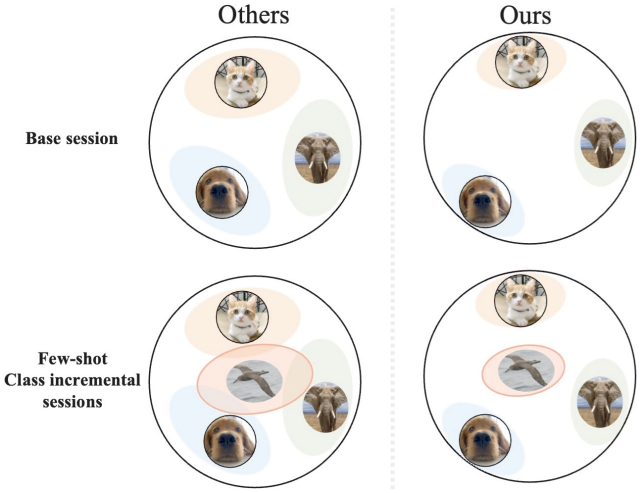

Figure 1: This is a schematic diagram of the boosting of our method in FSCIL. Our method can significantly reduce the intra-class distance, boost the inter-class distance, and have better spacing for the added classes. It is reflected in the accuracy results and the visual distribution of the final features, detailed analysis in Table 1 and Figure 4.

*Corresponding author

method called Multiple Mixing Self-Distillation(M2SD) for FSCIL, which prepares the feature space for incremental sessions and emphasizes feature extensibility and inclusiveness in later incremental sessions, showed in Figure 1.

To prepare the feature space for subsequent incremental sessions, M2SD utilizes multi-scale feature extraction and fusion; the specific structure can be seen in Figure 2. We extract features at different scales for each instance and fuse them to capture the characteristics of multiple dimensions of the instances fully. By comparing the fusion of these features with a single feature, as in previous work(Ji et al. 2021), the feature module can better understand various aspects of the instance, leading to greater inclusiveness. For expansion, inspired by using virtual classes in FACT, we propose a dual-branch virtual class to enhance further the extensibility of the feature extraction module(Xu and Liu 2019). It becomes aware of potential expansion possibilities by optimizing the dual-branch virtual class and allowing the feature module to see the unknown in advance. It tries to reserve feature space for new classes in the future.

Given that the entire training phase is divided into base and incremental sessions, and referring to CEC(Zhang et al. 2021) to address the issue of imbalanced data between old and new classes, we decouple the model into feature and classifier learning, which can be seen by Figure 2. This structure is used in many CEC follow-up works(Zhou et al. 2022; Song et al. 2023). The parameters are frozen once the feature module is learned using sufficient data from the base session. The feature module functions solely as a feature extractor during incremental learning, and its parameters are no longer updated. The classifier is the only part of the model updated upon the arrival of new classes.

The three main contributions of our paper:

- To our knowledge, we are the first to introduce self-distillation to FSCIL. Our method improves model feature discriminative and classification accuracy.
- We propose a two-branch virtual inter-class distillation. Based on the successful use of mixup(Zhang et al. 2017) to construct virtual classes, we have successfully introduced the virtual class constructed by CutMix(Yun et al. 2019), which was negative for FSCIL tasks.
- Experiments on CIFAR100, CUB200, and miniImageNet demonstrate that our method outperforms the baseline and achieves new SOTA results.

Related Work

Few-Shot Learning(FSL)

Few-Shot Learning (FSL) refers to the task of training models on small datasets that typically have only a few instances per class. The goal of it is to overcome the challenge of overfitting in traditional deep learning methods, which require large amounts of data to train. Extensive related work in this field includes metric-based methods(Snell, Swersky, and Zemel 2017a) such as prototype-based models, and optimization-based methods(Li et al. 2019) like gradient-based optimization. Another popular approach is to use meta-learning(Snell, Swersky, and Zemel 2017b), where the model learns how to learn from few-shot instances by training on various tasks. Recent work(Tian et al. 2020; Zhang et al. 2023b) demonstrates that a good feature space is essential for FSL, and further enhancements can be achieved by incorporating self-distillation. Inspired by its findings, we applied self-distillation to FSCIL to obtain a high-quality feature space.

Class-Incremental Learning(CIL)

Class-Incremental Learning (CIL) refers to the problem of continuously learning new classes in a lifelong learning scenario, where the data distribution of the new classes may differ from the previously learned classes. Various approaches have been proposed in the literature, including regularization-based methods(Li and Hoiem 2017), memory-based methods(Lopez-Paz and Ranzato 2017; Gidaris and Komodakis 2018), and architecture-based methods(Rusu et al. 2016). In particular, IL2A(Zhu et al. 2021a) highlights the critical role of representation learning in CIL. Integrating data augmentation and knowledge distillation used in IL2A inspires our research.

Few-Shot Class-Incremental Learning(FSCIL)

Few-shot Class-incremental learning(FSCIL) builds upon the foundations of FSL and CIL(Akyürek et al. 2021). As a result, the category(Kim and Choi 2021) of FSCIL methods is not fundamentally different from them. Prior works in this area include CEC, which decouples the model into two parts, representation learning, and classifier learning, and it is well-suited for FSCIL. FACT addresses the FSCIL challenge by constructing virtual classes using manifold mixup(Verma et al. 2019), which improves the quality of the trained feature space for the incremental sessions. Both of these ideas inspired our method.

Knowledge Distillation

Knowledge distillation(Hinton, Vinyals, and Dean 2015) is a method that involves transferring knowledge from a pre-trained model, known as the teacher model, to a new model, known as the student model, to improve student performance. The student model may have a different architecture or be smaller than the teacher model, but they do not need to be identical. Self-distillation(Lee, Hwang, and Shin 2020; Zhang et al. 2019) is a specific case of knowledge distillation in which a single model generates soft targets that are then used to train another identical model. In this paper, we are the first to incorporate self-distillation into FSCIL. It can facilitate the model to learn that the feature space will be enriched and that knowledge can be internalized, ultimately leading to enhanced model generalization. So we design a Multiple Mixing Self-Distillation(M2SD) into FSCIL to get a feature space more suitable for FSCIL.

Problem Set-Up

FSCIL comprises multiple learning sessions that are completed sequentially. During each session, the model is extended to a new subset of the dataset, while the training subsets from previous sessions are inaccessible. The evaluation

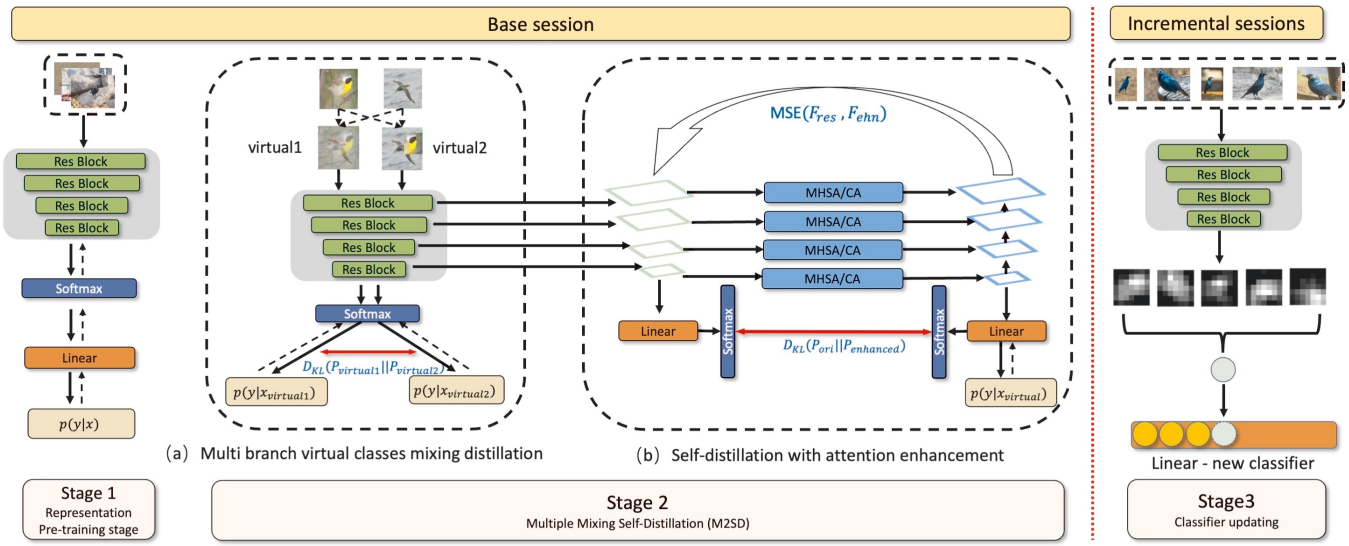

Figure 2: Our methodology framework is divided into two main parts: base session and incremental sessions. Base session are divided into two stages. One is the general model pre-training stage, and the other is the stage of M2SD, which is composed of two self-distillation modules. Incremental sessions have only one stage, which is classifier updating.

of the FSCIL method in each session considers all classes learned in previous and current sessions. In summary, the overall training process is divided into a base session and multiple incremental sessions.

Base Session: At this session, the model receives sufficient training data from the training set $\mathcal{D}^0_{train} = (x_i, y_i)^{n_{train}}_{i=1}$ and is then evaluated on the test set $\mathcal{D}^0_{test} = (x_i, y_i)^{n_{test}}_{i=1}$. x_i is an instance in the training dataset, and its corresponding label $y_i \in Y_0$. Y_0 is the label space of $\mathcal{D}^0$.

Incremental Sessions: New classes arrive incrementally with insufficient instances at these sessions. A series of datasets $\{\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^B\}$ appearing in sequence. B means there are a total of B incremental sessions. $\mathcal{D}^b = (x_i, y_i)^{NK}_{i=1}$, $y_i \in Y_b$ which is the label space of session b and $Y_b \cap Y_{b'} = \varnothing$ for $b \neq b'$. At session b, only dataset $\mathcal{D}^b$ can be reached. The subset is represented as an N-way, K-shot to form, which denotes that each session dataset is wrapped in N classes, each class with K instances. After training the dataset for a new session, the model should test the model with classes from this session and all previous sessions.

Method

In this section, we present our framework to solve the FSCIL problem. Our method consists of three stages: the first two are performed in the base session, and the last stage for the incremental sessions. The central part of our method is in Stage2. In the optimal setting, it accounts for the latter 80% epochs of the base session training, and it can be explained in subsequent ablation experiments, shown in Table 3.

Representation Pre-training Stage

This stage begins the base session, and the methodology follows general deep learning. Given the abundance of classes $\{y_1, ..., y_i\} \in Y_0$, and corresponding instances $\{x_1, ..., x_i\} \in X_0$ from the dataset $\mathcal{D}^0$. This stage focuses on enabling the model to quickly assimilate the dataset's characteristics and acquire the most suitable feature space for the next stage. Our model is $f_\theta(x)$ with a backbone $\phi(x)$ and a linear classifier W.

$$f_\theta(x) = W^T \phi(x) \tag{1}$$

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} L_{CE}(y_i, W^T \phi(x_i)) \tag{2}$$

where θ represents $\phi(x)$ and W parameters. m is the number of training samples, x_i is the i-th input instance from X_0, y_i is the corresponding label from base session label space Y_0.

This stage is crucial in enabling successful self-distillation operations in the subsequent stages. Without a suitable pre-trained backbone network $f_\theta(x)$, models initialized with random parameters can only impart random knowledge, rendering them ineffective in supporting successful distillation. We will demonstrate the importance of this stage and its impact on the results in the subsequent ablation experiments, shown in Table 3.

Multi Branch Virtual Classes Mixing Distillation

SAVC(Song et al. 2023), FACT(Zhou et al. 2022), and IL2A(Zhu et al. 2021a) have both improved future predictability by using virtual classes. Based on these works, we have made specific enhancements to strengthen further the expanding effect of this virtual class on the inter-class distance within the feature space. By introducing multi-instance, we construct virtual classes from different ensemble perspectives, enabling the model to learn virtual classes that can keep feature space for future insertion of new classes.

For our multi-branch, we utilize two different methods: mixup(Zhang et al. 2017) and CutMix(Yun et al. 2019).

Method	Accuracy in each session (%)								
	0	1	2	3	4	5	6	7	8
Finetune	61.31	27.22	16.37	6.08	2.54	1.56	1.93	2.60	1.40
iCarl(Rebuffi et al. 2017)	61.31	46.32	42.94	37.63	30.49	24.00	20.89	18.80	17.21
NCM(Hou et al. 2019)	61.31	47.80	39.31	31.91	25.68	21.35	18.67	17.24	14.17
TOPIC(Tao et al. 2020)	61.31	50.09	45.17	41.16	37.48	35.52	32.19	29.46	24.42
CEC†(Zhang et al. 2021)	72.23	66.86	63.19	60.00	57.20	54.31	51.62	49.92	48.28
ERDFR(Liu et al. 2022)	71.84	67.12	63.21	59.77	57.01	53.95	51.55	49.52	48.21
MetaFSCIL(Chi et al. 2022)	72.04	67.94	63.77	60.29	57.58	55.16	52.90	50.79	49.19
Fact†(Zhou et al. 2022)	76.30	71.00	66.61	62.84	59.41	56.06	53.10	51.06	49.31
ALICE(Peng et al. 2022)	80.60	70.60	67.40	64.50	62.50	60.00	57.80	56.80	55.70
SSFE-Net(Pan et al. 2023)	72.06	66.17	62.25	59.74	56.36	53.85	51.96	49.55	47.73
WaRP(Kim et al. 2022)	72.99	68.10	64.31	61.30	58.64	56.08	53.40	51.72	50.65
CABD(Zhao et al. 2023)	74.65	70.43	66.29	62.77	60.75	57.24	54.79	53.65	52.22
GKEAL(Zhuang et al. 2023)	73.59	68.90	65.33	62.29	59.39	56.70	54.20	52.59	51.31
SAVC†(Song et al. 2023)	81.02	75.59	71.36	67.48	64.40	60.86	57.97	56.10	54.59
Ours	**82.11**	**79.92**	**75.44**	**71.31**	**68.29**	**64.32**	**61.13**	**58.64**	**56.51**

Table 1: This table mainly shows the performance of our method and previous work on each session on the miniImageNet. Three of these methods†are based on our replication implementation, and the others are taken from their articles. For the other two datasets, refer to the Appendix for specific results.

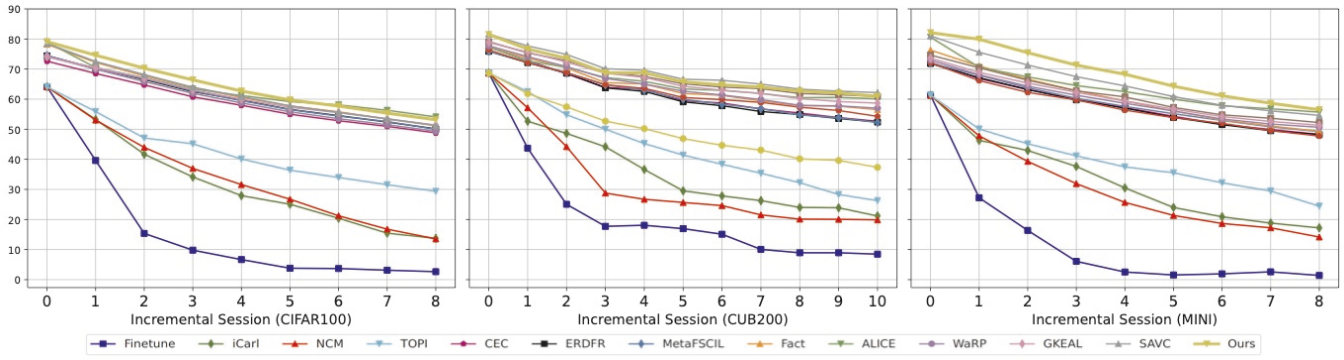

Figure 3: The comparison is made with the previous state-of-the-art works on three mainstream benchmark datasets in CIFAR100, CUB200, and miniImageNet. Our method outperforms all current methods for all sessions and achieves significant performance improvement in the average case. Detailed figures are in Table 1 and the Appendix.

Mixup generates virtual instances by linearly interpolating pairs of instances, focusing on producing greater diversity, which is crucial for virtual classes. In contrast, CutMix combines pairs of images by cutting and pasting, concentrating on generating more realistic instances. Since the new classes in the incremental sessions are real, the virtual classes need to be authentic. The diverse and realistic branches are distilled based on the characteristics of the newly arrived class instances in the incremental sessions. We must admit that not every virtual instance matches our analysis above, adding random factors to the method and making it more robust.

This stage is still part of the base session, so the dataset used is D_0. During the same mini-batch $\{x_1, ..., x_b\}$, b indicates the size of the mini-batch. We randomly interpolate two different classes of instances (x_i, x_j), corresponding $y_i \neq y_j$ to generate two instances of two virtual classes denoted as $x_I^{virtual}$ and $x_{II}^{virtual}$.

$$x_I^{virtual} = mixup(x_i, x_j) = \lambda x_i + (1-\lambda)x_j \quad (3)$$

$$x_{II}^{virtual} = CutMix(x_i, x_j) = M \odot x_i + (1-M) \odot x_j \quad (4)$$

$$\lambda \in \beta(\alpha, \alpha) \quad (5)$$

where two λ is the same, a random number of interpolation coefficients sampled from Beta distribution. $M \in \{0,1\}^{W \times H}$ is a binary mask indicating the patch's location cropped from the two instances. In order to get the CutMix mask, first determine the coordinates of the bounding box (r_x, r_y, r_w, r_h). Unif refers to Uniform Distribution.

$$\begin{aligned} r_x &\sim \text{Unif}(0 + r_w/2, W - r_w/2), \quad r_w = W\sqrt{1-\lambda} \\ r_y &\sim \text{Unif}(0 + r_h/2, H - r_h/2), \quad r_h = H\sqrt{1-\lambda} \end{aligned} \quad (6)$$

The corresponding labels of the virtual classes are $y_I^{virtual}$ and $y_{II}^{virtual}$.

$$y_I^{virtual} = y_{II}^{virtual} = \lambda y_i + (1-\lambda) y_j \tag{7}$$

Similar to other works(Song et al. 2023; Zhou et al. 2022) that use virtual classes, we force $f_\theta(x^{\text{virtual}})$ to fit the distribution of virtual classes with equation(8). The virtual class information is not directly used in the subsequent incremental sessions. However, it encourages the $f_\theta(x)$ to adapt to types that have never been seen before, a characteristic of incremental sessions.

$$\mathcal{L}_{ce}^{v} = \sum \mathcal{L}_{ce}(y_i^{virtual}, f_\theta(x_i^{virtual})), i = I, II \tag{8}$$

It is also required that the distribution obtained from the two virtual classes is consistent, which makes $f_\theta(x)$ stable and consistent in the virtual classes space composed of multi-mixture enhancements. So Kullback–Leibler divergence (Kullback and Leibler 1951) is used here.

$$\begin{aligned} \mathcal{L}_{KL}^{v} &= \mathcal{D}_{KL}\left(f_\theta(x_I^{virtual}) || (f_\theta(x_{II}^{virtual})\right) \\ &+ \mathcal{D}_{KL}\left(f_\theta(x_{II}^{virtual}) || (f_\theta(x_I^{virtual})\right) \end{aligned} \tag{9}$$

Self-Distillation with Attention Enhancement

In order to fuse features from multiple phases of the network and obtain more discriminative features(Zhang et al. 2023a), we designed a feature enhancement module $FE(x)$ to perform self-distillation on the network.

$$FE(x) = FPN(EN(\phi(x))) = FPN(\phi'(x))) \tag{10}$$

The first step is to use attention $EN(\cdot)$ to augment the feature obtained from $\phi(x)$ and obtain high-quality features $\phi'(x)$. Then multi-scale feature fusion is introduced to obtain more semantic information by FPN from the attention-enhanced virtual class features. Finally, we apply a self-distillation module that uses the enhanced features to align the original network features.

Attention Enhanced: To fully use the feature information of inputs, we use the attention mechanism to enhance the feature of each block in the $\phi(x)$, which also helps to reduce irrelevant information.

$$\phi(x) = [\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x)] \tag{11}$$

Each block $\phi_i(x)$ is enhanced by an attention mechanism. Multi-Head Self-Attention(MHSA)(Vaswani et al. 2017) is used for the first and last layer, and Coordinate Attention(CA)(Hou, Zhou, and Feng 2021) is used for the second and third. Using MHSA in the first block can provide a strong foundation for later blocks to build upon, which is particularly useful when the input data is complex. This is the biggest characteristic of virtual classes. In the second and third blocks, the CA selectively attends to different spatial locations within the feature maps, which virtual classes lack. Finally, in the fourth block, the network typically reverts to using MHSA, which can further refine and integrate the feature learned in the previous blocks. After attention enhancing, the feature $\phi_i'(x)$, i denotes the i-th block.

$$\phi_i'(x) = EN(\phi_i(x)) = MHSA/CA(\phi_i(x)) \tag{12}$$

Feature Fusion: To align the above feature after attention enhancement and leverage multi-scale information of virtual class features, we use a BiFPN(Tan, Pang, and Le 2019) structure to fuse them.

$$T_i = FPN(\phi_i'(x)) \tag{13}$$

T_i is attention-enhanced refined feature-map, i denotes the i-th block. Then, a temporary classifier $W^{'}$ is used to predict the result.

$$\hat{p} = W^{'}(T_{\text{last}}) \tag{14}$$

$\hat{p}$ is treated as a soft label.

Attention Transfer Loss: Finally, we introduce an Attention Transfer Loss(Zagoruyko and Komodakis 2016) to distillate the information between the original activation's attention map and the multi-scale attention-enhanced activation's attention map generated by $FE(\cdot)$.

$$\mathcal{L}_{\text{AT}} = D_{KL}\left(\hat{p}, f_\theta(x)\right) + \sum_{i=1}^{N} \left\| \text{at}\left(T_i'\right) - \text{at}\left(\phi_i(x)\right) \right\|_2^2 \tag{15}$$

where $\text{at}(\cdot)$ is the attention map computed from the input feature. Note that the loss function consists of two terms: the first is the KL divergence between model output $f_\theta(x)$ and soft label $\hat{p}$ from feature fusion. The second term measures the difference between the attention maps of T_i and ϕ_i^{virtual}. N is the number of attention maps. Finally, the loss function for our second stage is

$$\mathcal{L} = \mathcal{L}_{\text{ce}}^{\text{v}} + \mathcal{L}_{\text{KL}}^{\text{v}} + \mathcal{L}_{\text{AT}} \tag{16}$$

Classifier Updating Stage

This stage focuses on the incremental learning process. Similar to CEC, our method involves freezing the backbone network θ_ϕ after two stages of training in the base session, removing the FE structure, and only updating the classifier parameter W during subsequent incremental sessions. We consider a total of $(1 + B)$ sessions, where 1 denotes the base session and B denotes the B incremental sessions. To represent each class in these datasets, we average the data over all the instances corresponding to that class. We update the classifier by appending these class features to the weight matrix, i.e., $W_{\text{new}} = [W_{\text{new}}; \boldsymbol{w}_i, i \in Y^b]$.

$$\boldsymbol{w}_i = \frac{1}{K} \sum_{j=1}^{\left|\mathcal{D}^b\right|} \mathbb{I}\left(y_j = i\right) \phi_{f_\theta}(x_j) \tag{17}$$

$(x_j, y_j) \in \mathcal{D}^b$, $\mathcal{D}^b$ is the training dataset $\mathcal{D}^k, k \in \{0, 1, ..., B\}$ that contains instances belonging to class i. $\left|\mathcal{D}^b\right|$ is the number of instances in the $\mathcal{D}^b$. $\mathbb{I}\left(y_j = i\right)$ is an indicator function that is equal to 1 if the label of the j-th instance is i, and 0 otherwise. K is a normalization constant ensuring the prototype has the same scale as the representation vectors.

Experiment

In this section, we evaluate our method performance on three mainstream benchmark datasets: Caltech-UCSD Birds-200-2011 (CUB200) (Wah et al. 2011), CIFAR100 (Krizhevsky, Hinton et al. 2009), and miniImageNet (Russakovsky et al. 2015), the results are shown in Figure 3.

Metric	Method	Average value
Intra-class distance ↓	Baseline	1.35
	M2SD	0.98(-27%)
Inter-class distance ↑	Baseline	6.49
	M2SD	7.92(+22%)

Table 2: Based on feature vectors, our method reduces the average intra-class distance by 27% and increases the average inter-class distance by 22% compared to the baseline.

Implementation Details

Dataset: Following the guidance of CEC, we divide each dataset into base and incremental sessions. For CUB200, the base session comprises 100 classes, and each of the 10 incremental sessions comprises 100 classes. Each incremental session consists of 10 classes, with 5 instances per class. For CIFAR100 and miniImageNet, the base session comprises 60 classes, and each of the 8 incremental sessions comprises 40 classes. Each incremental session consists of 5 classes, with 5 instances per class.

Training Details: Our implementation is based on PyTorch, and the choice of backbone model is determined according to TOPIC(Tao et al. 2020): ResNet-20(He et al. 2016) for CIFAR100, ResNet-18 for CUB200 and miniImageNet. The same general data augmentation methods as other methods are used. We employ stochastic gradient descent (SGD) with momentum for optimization with a learning rate 1e-3 and a batch size of 256 on a 4xA100 GPU.

Benchmark Comparison

We mainly compare knowledge distillation-based methods and other SOTA methods. Our results on the three datasets are presented in Figure 3, showing that our method outperforms the current SOTA methods. Specifically, on the CUB200 our method achieves an average improvement of more than 2.0% over each stage compared to them. On the CIFAR100, the average improvement is more than 2.1%. On the miniImageNet, our method performs the best, which outperforms the SOTA method by more than 3.2% on average. Compared with the two SOTA methods based on knowledge distillation of SSFE-Net and CABD, our method is still very advantageous regarding accuracy across all sessions. Compared with them, our method improves by 7.21% and 8.94% per session on average.

Numerical Analysis

The effectiveness and characteristics of our method are reflected through the following numerical and visual analysis.

Visualization Increment: We used t-SNE(Van der Maaten and Hinton 2008) to visualize the change in the decision boundary on CIFAR100. Five classes were randomly selected from the base session, and another five classes were randomly selected from the incremental sessions. Figure 4a is the incremental result of the baseline method. The baseline method uses cross-entropy loss to train the entire base session, and then the incremental session uses our same classifier update strategy. Figure 4b is our method, and it can be seen that the added classes are more naturally entered into the feature space than the baseline method. Compared with the baseline, our method reduces the average intra-class distance by 27% and increases the average inter-class distance by 22%. The intra-class distance is the average distance from the same class of data to the centroid of the class. The inter-class distance is the minimum value between a class centroid and other class center centroids, shown in Table 2.

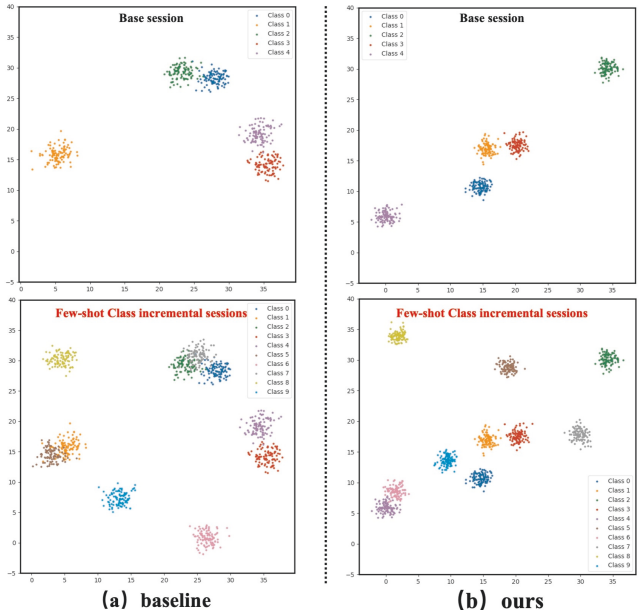

Figure 4: The distance coordinate scales of the two t-SNE are the same. Part (a) is the incremental result of the baseline method. The baseline method uses cross-entropy loss to train the entire base session, and then the incremental session uses our same classifier update strategy. Part (b) is the incremental result of our method.

Confusion Matrix: We plot the confusion matrix generated by training the model without(Figure 5a) and with our method(Figure 5b) on CIFAR100. As we can see, our diagonal is brighter, whether the base class is in the early stage or the incremental class in the later stage.

Nways-Kshots: We set the number of ways N and shots K in $\{1,5,10,15,20\}$. A total of 25 experiments are performed on the CUB200 and record the classification results of the last incremental session, shown in Figure 5c. It can be seen that the classifier update is insensitive to N for the same K. It is evident from the function of classifier update. When the same N, the larger K classification, the better the result, but the growth gradually decreases.

Ablation Study

The ablation experiment was set up in four parts, as shown in Table 3. The experiments were conducted on the CUB200 with other configuration parameters unchanged.

Division of Base Session: The primary purpose of the ex-

ablation	situation		Accuracy in each session (%)										
	stage1	stage2	0	1	2	3	4	5	6	7	8	9	10
ratio	0	100%	81.19	75.59	71.48	65.82	64.28	60.38	59.35	57.33	55.63	53.89	52.37
	50%	50%	81.59	77.19	74.08	**69.05**	68.90	65.78	**65.01**	**64.20**	62.13	61.83	60.73
	80%	20%	**81.87**	**77.46**	**74.25**	69.00	**68.97**	**66.03**	64.90	63.81	62.54	61.81	60.63
	100%	0	73.38	66.90	63.55	58.15	57.56	55.47	54.40	52.51	50.43	49.56	48.35
	20%	**80%**	81.49	76.67	73.58	68.77	68.73	65.78	64.73	64.03	**62.70**	**62.09**	**60.96**
	virtual1	virtual2	0	1	2	3	4	5	6	7	8	9	10
method	mixup	-	81.30	76.10	73.30	68.03	68.44	65.30	63.87	63.34	61.50	61.19	60.19
	CutMix	-	81.19	75.59	71.48	65.82	64.28	60.38	59.35	57.33	55.63	53.89	52.37
	mixup	mixup	**81.66**	**76.69**	**74.01**	**69.34**	68.25	65.19	63.85	62.51	61.64	60.56	59.49
	CutMix	CutMix	81.21	76.35	73.41	68.68	68.08	65.10	64.15	63.33	61.65	60.94	59.88
	mixup	**CutMix**	81.49	76.67	73.58	68.77	**68.73**	**65.78**	**64.73**	**64.03**	**62.70**	**62.09**	**60.96**
	FE	dual	0	1	2	3	4	5	6	7	8	9	10
FE	✓		80.88	75.78	72.19	67.55	67.32	64.22	63.51	62.73	61.81	60.87	60.02
		✓	81.25	76.25	72.92	68.51	68.12	64.78	63.68	62.93	61.70	60.85	59.84
	✓	✓	**81.49**	**76.67**	**73.58**	**68.77**	**68.73**	**65.78**	**64.73**	**64.03**	**62.70**	**62.09**	**60.96**
	updating		0	1	2	3	4	5	6	7	8	9	10
Classifier			81.49	72.81	66.57	61.32	56.83	53.00	49.57	46.60	43.96	42.36	40.17
	✓		**81.49**	**76.67**	**73.58**	**68.77**	**68.73**	**65.78**	**64.73**	**64.03**	**62.70**	**62.09**	**60.96**

Table 3: Ablation study on CUB200. The first column shows the different methods used. The second column provides specific details about the settings used for each method. The last row of each sub-table indicates the best-performing implementation selected in this study.

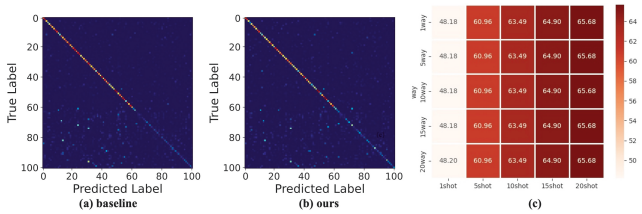

Figure 5: Part(a) and Part(b) shows that the effectiveness and accuracy of our method are improved. Part(c) primarily shows the impact of classifier updating, which is not sensitive to the incremental increase in the number of classes.

periments here is to demonstrate the necessity of the pre-train stage and its trade-off with our method in the base session.

We have set the ratio of pre-training and our method based on epochs. As shown in the first part of Table 3, it is clear that our method directly leads to the worst results without pre-training the model. The average performance is 0.5% lower in the base session and 8% lower in the final session. It can also be seen that the larger the ratio, the better the performance in the early stages, while the later performance gets worse. To average the overall performance, we choose the ratio of 20%:80%.

Multiple Mixing Self-Distillation: As shown in the first part of Table 3, when stage1 ratio is 100%, it is equivalent to not using our self-distillation method. Comparing with our final results, we can see that our method has an average increase of 10% per session. It can reflect the effectiveness of our self-distillation method.

Multi Branch Virtual Classes Mixing Distillation: As shown in the second part of Table 3, the ablation experiments for this aimed to investigate the necessity of dual branches and the choice of virtual class construction method for each branch. Using mixup alone to construct virtual classes yields good results. However, when using CutMix alone, the last session decreased by 8.4% compared to our method. The improvement in the base session through virtual classes constructed by CutMix is similar to the mixup. However, CutMix drops much faster in the incremental sessions than in the mixup. It shows that the feature space obtained by CutMix training is unsuitable for FSCIL. However, combining CutMix and mixup with distillation can further improve the gain of virtual class features brought by mixup to FSCIL while avoiding the negative impact of CutMix.

We also compared dual branches using the same virtual class construction method. The experimental involved sampling different lambda values and randomly selecting two different classes. It was found that no other combination is better than ours.

Self-Distillaton with Attention Enhancement: The results in the third part of Table 3 showed that the dual-branch the virtual class structure provided an average boost of 1.15% per session, while the feature enhancement struc-

ture provided an average boost of 0.79% per session. The attention enhancement module demonstrated the most significant improvement between the two experiments.

Classifier Learning: In the fourth part of the Table 3, we can see that if we do not empty the original W parameters directly into the incremental session classification, there is a significant impact on the subsequent classification effect. From session 1 lags behind 3.86%, the lagging magnitude increases until the last session lags behind 20.79%, with an average drop of 13.49% per session. It can be seen that the use of classifier updates is essential.

Conclusion

To solve the FSCIL problem, we first employ dual-branch virtual class distillation to expand the feature space. This expansion enables it to accommodate both current and future classes, which we have verified through numerical and visual methods. Furthermore, we employ feature enhancement and self-distillation to fully utilize the virtual class feature and enhance the compatibility of the feature space. These methods allow us to obtain a feature space suitable for the FSCIL. Experimental results on three datasets demonstrate that our approach leads to significant session accuracy gains, indicating the effectiveness of our M2SD.

Acknowledgments

This work was supported by Alibaba Group through Alibaba Research Intern Program. We thank the support from the Alibaba-South China University of Technology Joint Graduate Education Program. This work was also partially supported by Guangdong Artificial Intelligence and Digital Economy Laboratory (Guangzhou).

References

Akyürek, A. F.; Akyürek, E.; Wijaya, D. T.; and Andreas, J. 2021. Subspace regularizers for few-shot class incremental learning. *arXiv preprint arXiv:2110.07059*.

Ayub, A.; and Wagner, A. R. 2020. Cognitively-inspired model for incremental learning using a few examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 222–223.

Chen, W.-Y.; Liu, Y.-C.; Kira, Z.; Wang, Y.-C. F.; and Huang, J.-B. 2019. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*.

Chi, Z.; Gu, L.; Liu, H.; Wang, Y.; Yu, Y.; and Tang, J. 2022. MetaFSCIL: a meta-learning approach for few-shot class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14166–14175.

Gidaris, S.; and Komodakis, N. 2018. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4367–4375.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hou, Q.; Zhou, D.; and Feng, J. 2021. Coordinate Attention for Efficient Mobile Network Design. *Cornell University - arXiv*.

Hou, S.; Pan, X.; Loy, C. C.; Wang, Z.; and Lin, D. 2019. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 831–839.

Ji, M.; Shin, S.; Hwang, S.; Park, G.; and Moon, I.-C. 2021. Refine myself by teaching myself: Feature refinement via self-knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10664–10673.

Kim, D.-Y.; Han, D.-J.; Seo, J.; and Moon, J. 2022. Warping the space: Weight space rotation for class-incremental few-shot learning. In *The Eleventh International Conference on Learning Representations*.

Kim, J.-Y.; and Choi, D.-W. 2021. Split-and-bridge: Adaptable class incremental learning within a single neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8137–8145.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2017. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6): 84–90.

Kullback, S.; and Leibler, R. A. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1): 79–86.

Lee, H.; Hwang, S. J.; and Shin, J. 2020. Self-supervised label augmentation via input transformations. In *International Conference on Machine Learning*, 5714–5724. PMLR.

Li, A.; Luo, T.; Xiang, T.; Huang, W.; and Wang, L. 2019. Few-shot learning with global class representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9715–9724.

Li, Z.; and Hoiem, D. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12): 2935–2947.

Liu, H.; Gu, L.; Chi, Z.; Wang, Y.; Yu, Y.; Chen, J.; and Tang, J. 2022. Few-Shot Class-Incremental Learning via Entropy-Regularized Data-Free Replay.

Lopez-Paz, D.; and Ranzato, M. 2017. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30.

Pan, Z.; Yu, X.; Zhang, M.; and Gao, Y. 2023. SSFE-Net: Self-Supervised Feature Enhancement for Ultra-Fine-Grained Few-Shot Class Incremental Learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 6275–6284.

Peng, C.; Zhao, K.; Wang, T.; Li, M.; and Lovell, B. C. 2022. Few-Shot Class-Incremental Learning from an Open-Set Perspective. *Springer, Cham*.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2001–2010.

Ren, M.; Liao, R.; Fetaya, E.; and Zemel, R. 2019. Incremental few-shot learning with attention attractor networks. *Advances in neural information processing systems*, 32.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115: 211–252.

Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; and Hadsell, R. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Shi, G.; Chen, J.; Zhang, W.; Zhan, L.-M.; and Wu, X.-M. 2021. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in Neural Information Processing Systems*, 34: 6747–6761.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Snell, J.; Swersky, K.; and Zemel, R. 2017a. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Snell, J.; Swersky, K.; and Zemel, R. 2017b. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Song, Z.; Zhao, Y.; Shi, Y.; Peng, P.; Yuan, L.; and Tian, Y. 2023. Learning with Fantasy: Semantic-Aware Virtual Contrastive Constraint for Few-Shot Class-Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 24183–24192.

Tan, M.; Pang, R.; and Le, Q. V. 2019. EfficientDet: Scalable and Efficient Object Detection. *Cornell University - arXiv*.

Tao, X.; Hong, X.; Chang, X.; Dong, S.; Wei, X.; and Gong, Y. 2020. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12183–12192.

Tian, Y.; Wang, Y.; Krishnan, D.; Tenenbaum, J. B.; and Isola, P. 2020. Rethinking few-shot image classification: a good embedding is all you need? In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, 266–282. Springer.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. *Neural Information Processing Systems*.

Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Lopez-Paz, D.; and Bengio, Y. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International conference on machine learning*, 6438–6447. PMLR.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.

Wu, Y.; Chen, Y.; Wang, L.; Ye, Y.; Liu, Z.; Guo, Y.; and Fu, Y. 2019. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 374–382.

Xu, T.-B.; and Liu, C.-L. 2019. Data-distortion guided self-distillation for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5565–5572.

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. *Cornell University - arXiv*.

Zagoruyko, S.; and Komodakis, N. 2016. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*.

Zhang, C.; Song, N.; Lin, G.; Zheng, Y.; Pan, P.; and Xu, Y. 2021. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12455–12464.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond Empirical Risk Minimization. *Cornell University - arXiv*.

Zhang, J.; Gao, L.; Hao, B.; Huang, H.; Song, J.; and Shen, H. 2023a. From Global to Local: Multi-scale Out-of-distribution Detection. *IEEE Transactions on Image Processing*.

Zhang, J.; Gao, L.; Luo, X.; Shen, H.; and Song, J. 2023b. DETA: Denoised Task Adaptation for Few-Shot Learning. *arXiv preprint arXiv:2303.06315*.

Zhang, L.; Song, J.; Gao, A.; Chen, J.; Bao, C.; and Ma, K. 2019. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3713–3722.

Zhao, L.; Lu, J.; Xu, Y.; Cheng, Z.; Guo, D.; Niu, Y.; and Fang, X. 2023. Few-Shot Class-Incremental Learning via Class-Aware Bilateral Distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11838–11847.

Zhou, D.-W.; Wang, F.-Y.; Ye, H.-J.; Ma, L.; Pu, S.; and Zhan, D.-C. 2022. Forward compatible few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9046–9056.

Zhu, F.; Cheng, Z.; Zhang, X.-Y.; and Liu, C.-l. 2021a. Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems*, 34: 14306–14318.

Zhu, K.; Cao, Y.; Zhai, W.; Cheng, J.; and Zha, Z.-J. 2021b. Self-promoted prototype refinement for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6801–6810.

Zhuang, H.; Weng, Z.; He, R.; Lin, Z.; and Zeng, Z. 2023. 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 (CVPR)*, 7746–7755.