Neural Networks 190 (2025) 107608

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Full Length Article

Embedding Space Allocation with Angle-Norm Joint Classifiers for few-shot class-incremental learning

Dunwei Tu [a,b], Huiyu Yi [a,b], Tieyi Zhang [a,b], Ruotong Li [a,b], Furao Shen [a,b,*], Jian Zhao [c]

[a] National Key Laboratory for Novel Software Technology, Nanjing University, China
[b] School of Artificial Intelligence, Nanjing University, Nanjing, 210023, China
[c] School of Electronic Science and Engineering, Nanjing University, Nanjing, 210023, China

ARTICLE INFO

Keywords:
Few-shot learning
Incremental learning
Embedding space allocation
Prototype learning

ABSTRACT

Few-shot class-incremental learning (FSCIL) aims to continually learn new classes from only a few samples without forgetting previous ones, requiring intelligent agents to adapt to dynamic environments. FSCIL combines the characteristics and challenges of class-incremental learning and few-shot learning: (i) Current classes occupy the entire feature space, which is detrimental to learning new classes. (ii) The small number of samples in incremental rounds is insufficient for fully training. In existing mainstream virtual class methods, to address the challenge (i), they attempt to use virtual classes as placeholders. However, new classes may not necessarily align with the virtual classes. For challenge (ii), they replace trainable fully connected layers with Nearest Class Mean (NCM) classifiers based on cosine similarity, but NCM classifiers do not account for sample imbalance issues. To address these issues in previous methods, we propose the class-center guided embedding Space Allocation with Angle-Norm joint classifiers (SAAN) learning framework, which provides balanced space for all classes and leverages norm differences caused by sample imbalance to enhance classification criteria. Specifically, for challenge (i), SAAN divides the feature space into multiple subspaces and allocates a dedicated subspace for each session by guiding samples with the pre-set category centers. For challenge (ii), SAAN establishes a norm distribution for each class and generates angle-norm joint logits. Experiments demonstrate that SAAN can achieve state-of-the-art performance and it can be directly embedded into other SOTA methods as a plug-in, further enhancing their performance.

1. Introduction

Deep Neural Network (DNN) methods have excelled in the field of computer vision (Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009; He, Zhang, Ren, & Sun, 2016; Simonyan & Zisserman, 2014; Tan, Pang, & Le, 2020). Leveraging vast datasets and known classification targets, DNNs have demonstrated remarkable capabilities. However, in real-life scenarios, datasets are often limited in size, so the application of few-shot learning (FSL) is more prevalent (Snell, Swersky, & Zemel, 2017; Sung et al., 2018; Wang, Yao, Kwok, & Ni, 2020). Moreover, in many application contexts, not only is the number of samples limited, but the number of classes continues to grow. Constrained by computational resources and practical constraints, retraining the entire model from scratch with each new class addition is not feasible. Therefore, the design of efficient and accurate algorithms for achieving Few-Shot Class-Incremental Learning (FSCIL) has attached increasing attention (Kalla & Biswas, 2022; Lin, Wu, Lin, Huang, & Luo, 2024; Tao et al., 2020; Yang et al., 2023; Zhang et al., 2021; Zhou, Wang et al., 2022) (see Fig. 1).

The goal of FSCIL is to acquire new knowledge effectively while preventing the forgetting of previously learned knowledge. Previous research on FSCIL shows that models trained incrementally are prone to overfit on limited new data and forget the old knowledge catastrophically (Cheraghian et al., 2021; Dong et al., 2021; Tao et al., 2020). To combat overfitting and catastrophic forgetting, some studies propose the use of a feature extractor freezing paradigm (Kalla & Biswas, 2022; Song et al., 2023; Zhang et al., 2021; Zhou, Wang et al., 2022). This paradigm freezes most layers in the Convolutional Neural Network (CNN), which is the feature extractor, and uses the Nearest Class Mean (NCM) algorithm (Mensink, Verbeek, Perronnin, & Csurka, 2013) instead of fully connected layers as the classifier during the incremental phase. According to the neural collapse phenomenon (Han, Papyan, & Donoho, 2021; Kothapalli, 2022), researchers discover that without any intervention, current training samples would occupy the entire embedding space. This phenomenon is beneficial for enhancing the neural network's capabilities in conventional classification tasks. However, in

* Corresponding author at: School of Artificial Intelligence, Nanjing University, Nanjing, 210023, China.
E-mail addresses: tudunwei@smail.nju.edu.cn (D. Tu), 211300010@smail.nju.edu.cn (H. Yi), frshen@nju.edu.cn (F. Shen).

https://doi.org/10.1016/j.neunet.2025.107608
Received 11 November 2024; Received in revised form 14 March 2025; Accepted 9 May 2025
Available online 27 May 2025

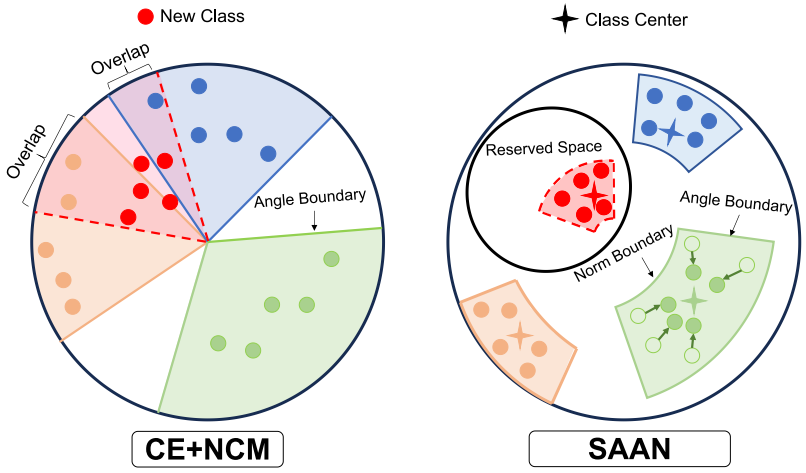


Fig. 1. The motivation of SAAN. In only cross-entropy supervision, old categories fill the entire feature space, making it hard for new categories to be inserted without overlapping. In SAAN, the old categories are allocated to a limited space, guided by the category centers, allowing new categories to be inserted into the reserved space without overlapping. In addition, compared to NCM, which has only angular decision boundaries, SAAN has both angular and norm boundaries, which fully utilizes the feature information to enhance the basis for discrimination.

the context of Class-Incremental Learning (CIL), learning new classes becomes extremely challenging. This phenomenon addresses the challenge (i) we aim to solve. To address this issue, some researchers (Song et al., 2023; Zhou, Wang et al., 2022) propose occupying some of the embedding space with virtual classes generated through data augmentation (Yang, Guo, Zhao & Shen, 2024; Yang, Shen, & Zhao, 2024). Then, during the incremental phase, the space occupied by these virtual classes can be used for inserting new classes. However, these methods cannot control the location of the space occupied by virtual classes, as the occupied positions are determined by the features of the virtual samples. Some researchers (Yang et al., 2023) propose using pre-allocated fixed prototypes to reduce the feature misalignment problem during the incremental process. However, this approach of completely fixed and randomly allocated prototypes ignores the intrinsic meaning of the samples, which may lead to a discrepancy between the semantic distance of the samples and their distance in the feature space.

Because of the FSL setting, the small number of samples in incremental sessions is insufficient for large-scale training, addressing the challenge (ii) we aim to solve. For choosing the neural network classifier, NCM (Guerriero, Caputo, & Mensink, 2018) does not rely on extensive training and is suitable for the FSCIL problem, which is wildly used in the mainstream methods (Song et al., 2023; Zhang et al., 2021; Zhou, Wang et al., 2022). NCM solely computes the distance between samples and the mean of each class to classify them, which does not rely on a large amount of data. Some researchers (Thongtan & Phienthrakul, 2019; Zhou, Ethayarajh et al., 2022) find that cosine similarity is well-suited for comparing embeddings in DNN. NCM based on cosine similarity has achieved good results in FSCIL (Song et al., 2023; Zhang et al., 2021; Zhou, Wang et al., 2022). In the FSL setting, we believe that leveraging as much sample information as possible during the incremental sessions is essential. Based on our experiments, we observe that in the FSCIL scenario, there is a significant difference in the number of samples across different categories, leading to variations in norms between categories. However, this norm information is not used by NCM.

To address these issues, we propose the class-center guided embedding Space Allocation with Angle-Norm joint classifiers (SAAN) learning framework. SAAN consists of the Class-Center guided embedding Space Allocation (CCSA) and the Angle-Norm Joint classifiers (ANJ). The motivation of SAAN is illustrated in Fig. 1.

CCSA is designed to address the challenge (i) posed by CIL. Specifically, CCSA divides the feature space into multiple subspaces and allocates different subspaces for different incremental sessions by setting class centers, using our designed cosine center loss. CCSA resolves two issues found in previous work. First, in prior virtual class methods (Song et al., 2023; Zhou, Wang et al., 2022), due to the lack of guidance for new classes, the reserved space for virtual classes may not necessarily be utilized by new classes. CCSA imposes explicit spatial constraints on new classes, ensuring that they enter the reserved space. Second, unlike methods with completely fixed prototype allocation (Ahmed, Kukleva, & Schiele, 2024; Yang et al., 2023), where random allocation may cause similar classes to be far apart in the feature space, SAAN enables the adjustment of class centers within the designated space to ensure that prototype similarity aligns with the samples' semantic information.

ANJ is a new classifier, designed to address the challenge (ii) posed by FSL. ANJ establishes a norm distribution for each class to estimate the norm logit and combines this with NCM to estimate the angular logit, ultimately generating joint logits. In this way, ANJ can leverage previously overlooked norm information, which is especially important for the information-scarce FSL setting.

Since SAAN is designed to address issues in previous virtual class methods, it can be directly integrated into these State-Of-The-Art (SOTA) methods as a plug-in. Experiments in both conventional and open-ended scenarios demonstrate that SAAN not only achieves performance comparable to SOTA but also further enhances the performance of existing SOTA methods.

The main contributions of this paper are summarized as follows:

(1) CCSA guides sample points into specific subspaces through our meticulously designed cosine center loss. This resolves two issues found in previous methods: new classes may fail to appear in the reserved feature space, and similar classes may be far apart in the feature space.

(2) Our experiments demonstrate the importance of norm information for classification in the context of FSCIL. ANJ effectively combines angular and norm information, fully utilizing the information contained within the limited samples.

(3) Experiments demonstrate that SAAN can achieve performance close to SOTA and further improve SOTA models' performance. Specifically, SAAN improves the final round accuracy by over 3% across three datasets and two methods.

2. Related works

Few-Shot Learning. FSL aims to adapt to unseen classes using limited training instances (Fu, Fu, & Jiang, 2021; Snell et al., 2017; Sung et al., 2018; Wang et al., 2020). The two prominent approaches are

optimization-based methods and metric-based methods. Optimization-based methods (Arnold, Iqbal, & Sha, 2021; Nichol, Achiam, & Schulman, 2018; Ravi & Larochelle, 2016; Wang, Cheng, Yu, Guo, & Zhang, 2019), situated within meta-learning frameworks, aim to enhance the acquisition of generalizable representations from limited data. Conversely, metric-based algorithms (Liu et al., 2020; Oreshkin, Rodríguez López, & Lacoste, 2018; Triantafillou, Zemel, & Urtasun, 2017; Wang et al., 2019, 2021; Zhang, Cai et al., 2020) leverage a pre-trained feature extraction backbone, utilizing distinct distance metrics to ascertain the similarity or dissimilarity between support and query instances. The separation of tasks into feature extraction and classification has proven effective in FSCIL.

Class-Incremental Learning. CIL is defined as the ongoing incorporation of new knowledge while retaining previously acquired knowledge (Masana et al., 2022; Mittal, Galesso, & Brox, 2021; Zhang, Zhang et al., 2020). Algorithms dedicated to CIL can be broadly divided into four principal categories. The first approach aims to modify the network's architecture to expand the model's capacity for accommodating new tasks (Wang et al., 2022; Yan, Xie, & He, 2021). The second approach focuses on enhancing the distinction between older and newer classes by selectively revisiting old samples (Rebuffi, Kolesnikov, Sperl, & Lampert, 2017; Wu et al., 2019; Zhao, Xiao, Gan, Zhang, & Xia, 2020). The third approach employs knowledge distillation to maintain the model's pre-existing knowledge (Hinton, Vinyals, & Dean, 2015; Li & Hoiem, 2017; Rajasegaran, Khan, Hayat, Khan, & Shah, 2020). The fourth approach evaluates the importance of each parameter to ensure critical ones remain unaltered (Aljundi, Babiloni, Elhoseiny, Rohrbach, & Tuytelaars, 2018; Liu et al., 2018; Zenke, Poole, & Ganguli, 2017). Direct application of CIL methodologies to FSCIL is hindered by overfitting issues associated with limited sample sizes.

Few-Shot Class-Incremental Learning. FSCIL diverges from CIL in its requirement for learning from limited samples for novel classes. FSCIL is primarily challenged by the catastrophic forgetting of previously learned classes and a propensity for overfitting to new classes. To enable the model to have forward-compatible learning capabilities, some researchers (Song et al., 2023; Zhou, Wang et al., 2022) create virtual classes to reserve space for future classes, while others (Ahmed et al., 2024; Yang et al., 2023) set fixed prototypes for all classes in advance. FACT (Ahmed et al., 2024; Zhou, Wang et al., 2022) proposes a forward-compatible training by assigning virtual prototypes to squeeze the embedding of known classes and reserving for new classes. Besides, FACT forecasts potential new categories and prepares for the updating process. The virtual prototypes enable the model to accommodate future updates, serving as proxies dispersed throughout the embedding space to enhance the classifier's performance during inference. SAVC (Song et al., 2023) innovates by distinguishing new classes from existing ones through the incorporation of virtual classes into supervised contrastive learning. The virtual classes, generated through predefined transformations, serve not only as placeholders for unseen categories within the representation space but also enrich it with diverse semantic information. These virtual class methods demonstrate the importance of reserving embedding space for new classes and enhancing the separability of embeddings.

NC-FSCIL (Yang et al., 2023) sets up classifiers with an Equiangular Tight Frame (ETF) structure that considers future classes, and it induces class alignment with the corresponding classifier to mitigate the classifier misalignment issue during the incremental process. OrCo (Ahmed et al., 2024) maximizes the boundaries between classes and reserves space for subsequent incremental data by introducing perturbed prototypes in the feature space and maintaining orthogonality between classes. These methods of pre-allocated prototypes highlight the importance of prior space allocation and preventing the misalignment of old categories in new sessions.

FACT occupies embedding space through virtual prototypes, while SAVC generates negative samples for contrastive supervised learning via data augmentation to preserve embedding space. Additionally, contrastive learning enhances the model's ability to generalize, which is crucial in FSL. NC-FSCIL assigns fixed prototypes to categories randomly before training and uses prototype alignment as the supervisory signal for the model. SAAN is a more flexible and semantically aligned learning strategy based on similarity estimation for spatial pre-allocation and dynamic adjustment of prototypes.

3. Preparatory work

3.1. Problem setting

In FSCIL, the model is fed with a sequence of training dataset $\{\mathcal{D}^t_{train}\}_{t=0}^{T}$, where $\mathcal{D}^t_{train}$ is the training set at the session t and T is the number of all sessions. $\mathcal{D}^t_{train} = \{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i$ is the ith training sample and y_i is the ith label. $\left|\mathcal{D}^t_{train}\right|$ denotes the number of the training samples at session t. Each session t has a distinct label space $\mathcal{Y}^t$, which means that $\mathcal{Y}^t \cap \mathcal{Y}^{t'} = \varnothing$ for any $t \neq t'$. $\left|\mathcal{Y}^t\right|$ denotes the number of the classes at session t. In the base session, $\left|\mathcal{D}^0_{train}\right|$ is equal to the number of all the samples whose labels $\in \mathcal{Y}^0$ in the dataset. For example, on CIFAR100 (Krizhevsky, Hinton, et al., 2009), there are 500 training samples for each class at the base session. However, at the incremental sessions, we only can access a limited number of training samples. The training data is organized in an N-way K-shot format which means there are N classes, and each class contains K training samples at each incremental session. For example, on CIFAR100, each incremental session includes 5 classes, with each class having 5 samples, meaning $N = 5$ and $K = 5$. At each session t, the model is tested on all test samples whose labels $\in (\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t)$.

3.2. The model framework

The whole model can be decoupled as a feature extractor and a classifier which follows (Kalla & Biswas, 2022; Song et al., 2023; Tao et al., 2020; Zhang et al., 2021; Zhou, Wang et al., 2022). We employ a CNN as the feature extractor, denoted by $f(\cdot)$. The output of this CNN, $\mathbf{e}_i = f(\mathbf{x}_i)$, serves as the basis for classification. In the feature extractor frozen paradigm, the CNN is fully trained at the base session and fine-tuned on the basis of freezing most of the layers in CNN at the incremental sessions. The NCM classifier recognizes samples by:

$$y^{\mathrm{p}}_{\mathbf{x}} = \arg\max_j \ \cos\langle \mathbf{e}, \boldsymbol{w}^t_j \rangle, \tag{1}$$

where $y^{\mathrm{p}}_{\mathbf{x}}$ is the predicted class, $\cos\langle\cdot,\cdot\rangle$ is cosine similarity function, and $\boldsymbol{w}^t_j$ is the representative point of the class j at the session t.

The representative point $\boldsymbol{w}^t_j$ in NCM classifier is calculated by:

$$\boldsymbol{w}^t_j = \frac{1}{n^t_j} \sum_{i=1}^{n^t_j} \left(\mathbf{e}_i \mathbb{I}(y_i = j)\right), \tag{2}$$

where n^t_j is the number of samples of class j and $\mathbb{I}(\cdot)$ is the indicator function. As indicated by Eq. (1), the NCM classifier, which classifies based on the nearest neighbor principle by measuring the cosine similarity between sample embeddings and representative points, completely disregards the norm information of the sample embeddings.

3.3. Learning of the baseline model

The optimization goal of the feature extractor freezing paradigm baseline is to minimize $\mathcal{L}_{ce}(\phi(\mathbf{x}), y)$, where $\phi(\mathbf{x}) = W^\top f(\mathbf{x})$ is the model composed of the feature extractor and the linear classifier, and $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss function. Here, $\phi(\mathbf{x}) \in \mathbb{R}^{|\mathcal{Y}|\times 1}$, $W \in \mathbb{R}^{d\times|\mathcal{Y}|}$, $f(\mathbf{x}) \in \mathbb{R}^{d\times 1}$, and d is the feature dimension. After the base session, most layers of the CNN are frozen and fine-tuned. The NCM classifier expands by $W = \{\boldsymbol{w}^0_1, \boldsymbol{w}^0_2, \ldots, \boldsymbol{w}^t_{|\mathcal{Y}^t|-1}, \boldsymbol{w}^t_{|\mathcal{Y}^t|}\}$. $\boldsymbol{w}^t$ is continuously expanded during the incremental process based on new data. Although the feature-freezing base approach is simple, it outperforms many FSCIL methods, making it a strong baseline model. The training and classification algorithm depicting the feature-freezing base approach is shown in Alg. 1.

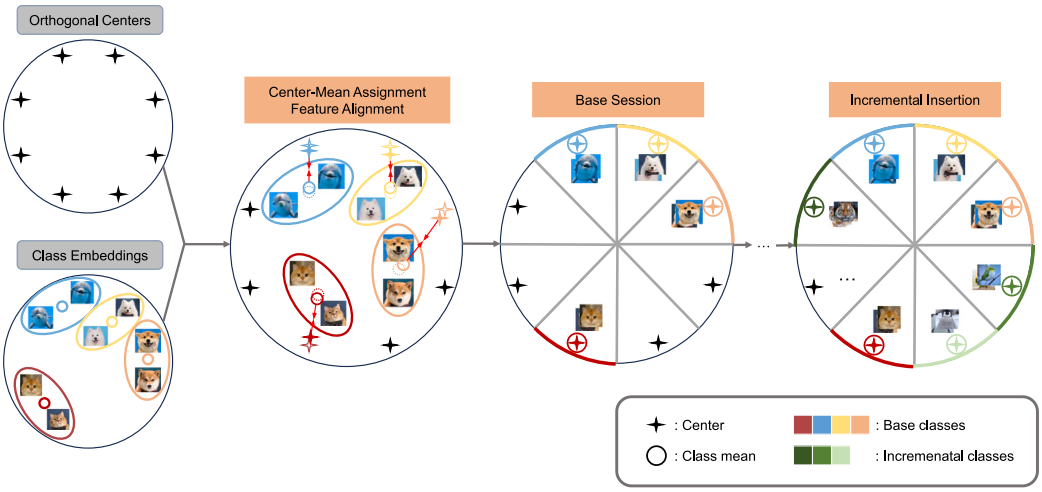


Fig. 2. Illustration of CCSA. CCSA sets orthogonal class centers and assigns these centers to different categories to allocate the embedding space. CCSA uses the well-designed cosine center loss to guide embeddings into the corresponding subspace, achieving the effect of feature alignment. The centers are adjusted based on semantics in the base session, and newly inserted classes are allocated to the remaining space during the incremental sessions.

Algorithm 1 The training and classification process for the feature-freezing base approach.

Require: The CNN $f(\cdot)$, The classifier W, The training set sequence $\{\mathcal{D}^t_{train}\}_{t=0}^{T}$, The model learning rate μ
Ensure: The updated CNN $f(\cdot)$, The updated classifier W

for each training dataset $\mathcal{D}^t_{train}$ **do**
  Expand the dimension of W from $d \times |\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^{t-1}|$ to $d \times |\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t|$
  if $t > 0$ **then**
    Freeze most layers of the CNN for fine-tuning preparation
  end if
  while $\mathcal{L}_{ce}$ not converges **do**
    for each batch $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\} \in \mathcal{D}^t_{train}$ **do**
      $e \leftarrow e + 1$
      Compute $\mathcal{L}_{ce}$ on $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$
      $\theta_f^{e+1} \leftarrow \theta_f^e - \mu \frac{\partial \mathcal{L}_{ce}}{\partial \theta_f^e}$
      $W^{e+1} \leftarrow W^e - \mu \frac{\partial \mathcal{L}_{ce}}{\partial W^e}$
    end for
  end while
  Recalculate W by Eq. (2) to replace fully connected layers by NCM
  Classify all the seen samples using the continuously expanding W and the continuously updating f through Eq. (1)
end for
return $f(\cdot)$, W

4. Method

The whole method, the class-center guided embedding Space Allocation with Angle-Norm joint classifiers (SAAN), consists of the Class-Center guided embedding Space Allocation (CCSA) and the Angle-Norm Joint classifiers (ANJ), which are illustrated in Figs. 2 and 3. CCSA and ANJ address the problems of guiding feature learning and classifier construction in FSCIL, respectively.

4.1. Class-center guided embedding space allocation

The baseline model trained solely with cross-entropy loss focuses only on the current sample during training and utilizes the entire embedding space, rendering it challenging for the model to assimilate new class knowledge (Han et al., 2021; Kothapalli, 2022). The embedding space consists of all possible feature vectors. This space not only needs to accommodate the feature vectors of the current samples but also needs to reserve capacity for future classes. Inspired by the center loss (Wen, Zhang, Li, & Qiao, 2016), we mitigate this challenge by establishing learnable class centers (prototypes) and guiding sample embeddings toward their corresponding class centers. Unlike center loss, CCSA constrains the allocation of center positions and the update of centers to achieve the goal of constraining the model's utilization of the embedding space and providing balanced space for new classes. The overall process of CCSA is shown in Fig. 2.

4.1.1. Space allocation

We first set $(|\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t|)$ orthogonal class centers and assign them one-to-one based on their distance from the class means. In the real world, considering the total number of classes is unknown and continuously growing, we just set d orthogonal centers, which is the maximum number of orthogonal centers that can be generated in a d-dimensional feature space. To ensure that the total distance between all class centers and their corresponding class means is minimized, we use the Hungarian algorithm (Kuhn, 1955; Munkres, 1957) to assign the class centers to different categories. The Hungarian algorithm is used to solve the assignment problem in a bipartite graph, aiming to find an optimal matching that minimizes the total cost. Here, the cost is defined by cosine distance, because the fully connected layer in a neural network performs dot product operations, and some literature (Goldberg & Levy, 2014; Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) suggests that dot products can introduce bias due to different sample sizes. Cosine distance, on the other hand, can eliminate this bias by removing the influence of vector norms. We define an $(|\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t|) \times (|\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t|)$ cost matrix $D = [c_{i,j}]$, where $c_{i,j}$ represents the cosine distance from i-th class center to the j-th class. Our objective is to find a one-to-one matching that minimizes the total cost. Mathematically, this can be written as:

$$\min_{\sigma} \sum_{i=1}^{n} c_{i,\sigma(i)},$$

where σ is a permutation of the set $\{1, 2, \ldots, (|\mathcal{Y}^0 \cup \mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t|)\}$, ensuring that each class center is uniquely assigned to a class and each class is uniquely assigned to a class center. Since the number of class centers in the base round exceeds the number of existing categories, we set up some virtual classes and assign a distance of 0 between these virtual class means and any class center to prioritize the allocation of existing categories. After the generation and assignment of centers, sample features are also distributed into different spaces, ultimately achieving the goal of spatial allocation.

4.1.2. Feature alignment

We design a novel cosine center loss to guide sample embeddings closer to their corresponding class centers and away from other class centers in the angular space, achieving the effect of feature alignment. We define the cosine center loss $\mathcal{L}_{cc}$ as $\alpha\mathcal{L}_1 + \beta\mathcal{L}_2$, where $\mathcal{L}_1$ pulls embeddings closer to their corresponding centers, $\mathcal{L}_2$ pushes embeddings away from other class centers, and α and β are weighting coefficients. $\mathcal{L}_1$ is defined by:

$$\mathcal{L}_1 = \frac{1}{m}\sum_{i=1}^{m}\left(1 - \cos\langle \mathbf{e}_i, \mathbf{c}_{y_i}\rangle\right), \tag{3}$$

where m is the batch size, and $\mathbf{c}_{y_i}$ is the class center of class y_i. $\mathcal{L}_2$ is defined by:

$$\mathcal{L}_2 = \frac{1}{m}\frac{1}{|\mathcal{Y}|}\sum_{i=1}^{m}\sum_{j=1}^{|\mathcal{Y}|}\left(\cos\langle \mathbf{e}_i, \mathbf{c}_j\rangle \mathbb{I}\left(y_i \neq j\right)\right). \tag{4}$$

$\mathcal{L}_1$ optimizes the cosine distance between sample embeddings and their corresponding class centers, restricting embeddings near the class centers to achieve the goal of space allocation. $\mathcal{L}_2$ increases the cosine distance between sample embeddings and centers of other classes. In addition to space allocation, $\mathcal{L}_{cc}$ also plays a role in reducing intra-class distance and improving inter-class separation. The detailed gradient analysis is provided in Appendix A. Under the guidance of class centers, the sample features will enter specific spaces, while retaining some space for the insertion of new classes.

4.1.3. Center updates

The pre-assigned and fixed centers may not semantically match the classes. In other words, semantically close centers may be far apart in embedding space. To ensure the semantic coherence of the centers, we combine CE loss and cosine center loss to jointly guide the model and adjust the centers during the learning process. The total loss is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{ce}(\phi(\mathbf{x}), y) + \alpha\mathcal{L}_1 + \beta\mathcal{L}_2. \tag{5}$$

To prevent unreasonable fixed center settings, we apply momentum adjustments to the centers by:

$$\mathbf{c}_j^{e+1} = \mathbf{c}_j^{e} + \eta\Delta\mathbf{c}_j, \tag{6}$$

where η is the center adjustment speed, and $\Delta\mathbf{c}_j$ is the momentum update. $\Delta\mathbf{c}_j$ is obtained by:

$$\arg\max_{\Delta\mathbf{c}_j}\sum_{i=1}^{m}\cos\langle \mathbf{e}_i, \Delta\mathbf{c}_j\rangle\mathbb{I}(y_i = j). \tag{7}$$

$\Delta\mathbf{c}_j$ is the point closest to all samples of the same class in terms of cosine distance. We take a simple solution of $\Delta\mathbf{c}_j$ as:

$$\Delta\mathbf{c}_j = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{\mathbf{e}}_i\mathbb{I}(y_i = j)\right), \tag{8}$$

where $\hat{\mathbf{e}}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|}$. Continuously adding the mean of normalized embeddings to the center will gradually move the center closer to the sample embeddings which is learned by cross-entropy loss. Setting the centers as learnable parameters and updating them through gradient descent is equivalent to using Eq. (6), as shown in Appendix A.

Allowing the centers to be updated freely, as in center loss, would compromise the spatial allocation function. Therefore, we let η decay rapidly after each epoch to ensure limited adjustment of the centers.

4.1.4. Incremental sessions insertion

In the incremental sessions, we continue to use the Hungarian algorithm to assign the class means of the current session to the unused class centers, and we reset η so that the centers can move within limited bounds. It should be noted that the already assigned class centers and old class means need to be removed, and the virtual class centers for future classes are retained. Due to the scarcity of incremental samples, only $\mathcal{L}_1$, which restricts the space, is activated, while $\mathcal{L}_2$, which improves separation, is deactivated.

4.2. Angle-norm joint classifiers

We aim to explore the knowledge embedded in the norms, which is ignored by NCM, by establishing a more comprehensive classifier that leverages both angular and norm information simultaneously.

4.2.1. Log-norm distribution of embeddings

NCM is based on cosine similarity, so it is an algorithm that relies on the angular relationship between vectors while ignoring the vector norm (the distance between the vector's tip and the origin). We attempt to incorporate this vector length information into the classification criteria. If the norms of samples from different categories have distinct distributions, it indicates that norms contain label information to a certain extent. The distributions of $\ln\|\mathbf{e}\|$ from different sessions and categories, as extracted by a CNN trained in the FSCIL paradigm, are shown in Fig. 4. In Figs. 4(a) and 4(c), it can be observed that the mean and variance of embedding log norms during the base sessions are significantly higher than those during the incremental sessions. Based on the characteristic demonstrated by the embedding log norm, we attempt to leverage this feature to assist us in determining whether this class belongs to the base session or the incremental sessions during inference. Additionally, as shown in Figs. 4(b) and 4(d), in the base session, the log-norm distributions of different classes vary significantly, whereas in the incremental sessions, this difference becomes much smaller. We believe that the variation in log-norm distributions may be caused by the model's varying degrees of fit to different categories. Although there is significant overlap in the norms across different categories and sessions, preventing direct prediction based solely on norms, norms can still serve as a part of the basis for classification.

4.2.2. Dual-granularity parametric estimation

As shown in Fig. 4, we observe significant differences between the incremental and base sessions, but the distributions across different incremental sessions are approximately similar. Based on this phenomenon, we choose to estimate a distribution for the norms of each class. Based on the trend of class distributions in Figs. 4(b) and 4(d), we assume that the log norms of the embeddings for samples belonging to the same category follow a normal distribution, represented as $\ln\|\mathbf{e}_i\| \sim \mathcal{N}(\mu_j, \sigma_j^2)$, where $y_i = j$, j is the category label, μ_j denotes the mean, and σ_j^2 denotes the variance.

During the base session, with sufficient training samples, we approximate the population mean and variance with sample mean and variance, respectively. As shown in Fig. 4(b), the differences in log norms between classes are significant, making it necessary to perform class-level modeling of the log norms. During the incremental sessions, however, each class contains only a few samples, which may result in significant deviations between the sample mean and variance and the true population values. Despite this, as shown in Fig. 4(d), the differences in log norm between classes during these sessions are actually quite small. We assume that the log norm of each incremental category follows the same normal distribution. We know that if the distribution of samples in each dataset follows the same normal distribution, then mixing the datasets will result in the samples still following a normal distribution, denoted as $\ln\|\mathbf{e}_i\| \sim \mathcal{N}(\mu_{\text{incre}}, \sigma_{\text{incre}}^2)$, where $y_i \in (\mathcal{Y}^1 \cup \cdots \cup \mathcal{Y}^t)$, $t \geq 1$, and $\mu_{\text{incre}}, \sigma_{\text{incre}}^2$ are the shared mean and variance, respectively. We conduct a test in Section 5.5 to determine whether the classes follow a normal distribution. We refer to assigning the same distribution to different classes in the incremental sessions as the shared distribution technique. A session-granular distribution does not directly improve classification accuracy. Its primary purpose is to distinguish between the base session and incremental sessions. By improving the accuracy of session identification, it indirectly enhances the final classification accuracy. In summary, we model the distributions of the base and incremental rounds with different levels of granularity.

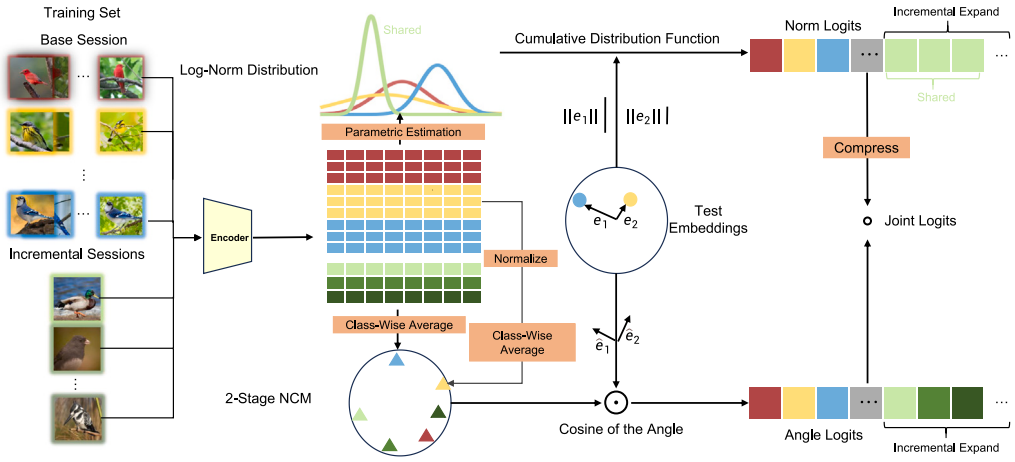


Fig. 3. Overall pipeline of ANJ. ANJ consists 2SNCM and Norm Distribution. 2SNCM is fairer than NCM in determining representative points, while Norm Distribution utilizes norm information to establish a probability model, enhancing the basis for classification.

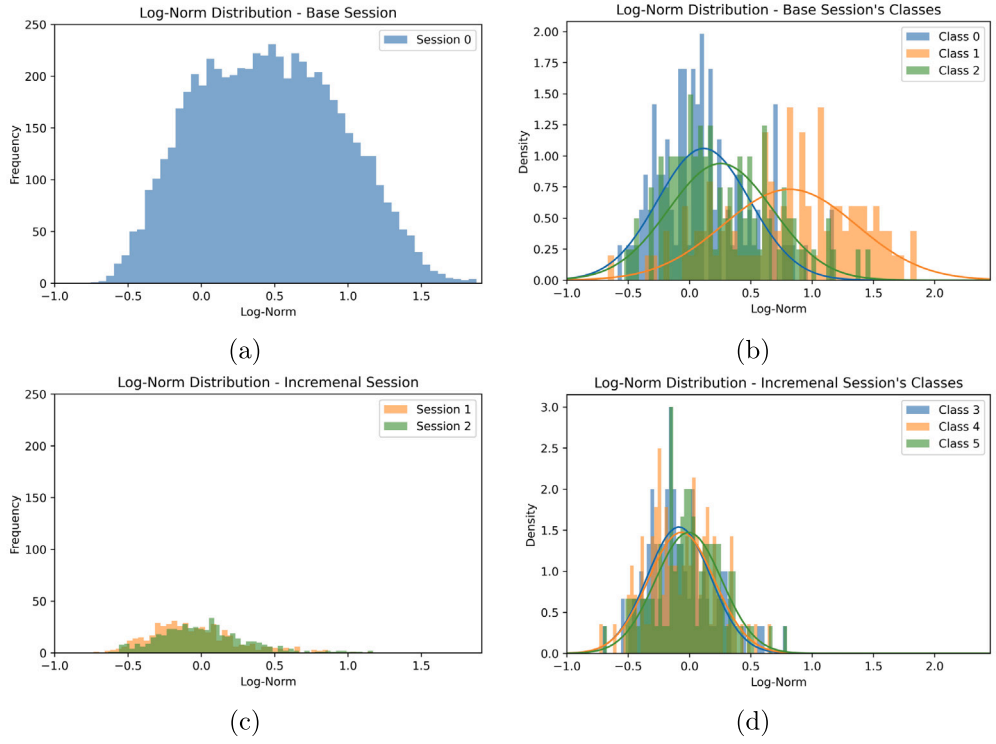


Fig. 4. Histogram of the distribution of the log norms of sample embedding on *mini*ImageNet dataset. (a) The base session. (b) 2 incremental sessions. (c) 3 randomly selected classes in the base session. (d) 3 randomly selected classes in incremental sessions. The curve represents the normal distribution formed by the sample mean and variance of the log norms.

4.2.3. Angle-Norm joint logits inference

The pipeline of Angle-Norm joint (ANJ) inference is illustrated in Fig. 3. ANJ consists of 2-Stage NCM (2SNCM) and Norm Distribution.

2-Stage NCM. We propose a variant of NCM, named 2-Stage NCM (2SNCM), to utilize angular information. For 2SNCM, the logits for the jth class $\mathbf{z}_j^1$ are given by $\mathbf{z}_j^1 = \cos\langle \mathbf{e}, \boldsymbol{w}_j \rangle$, which is consistent with NCM. The $\boldsymbol{w}_j$ is stage-dependent and is calculated by:

$$\boldsymbol{w}_j^t = \begin{cases} \frac{1}{n_j^t} \sum_{i=1}^{n_j^t} \left(\hat{\mathbf{e}}_i \mathbb{I}(y_i = j) \right) & \text{if } t = 0 \\ \frac{1}{n_j^t} \sum_{i=1}^{n_j^t} \left(\mathbf{e}_i \mathbb{I}(y_i = j) \right) & \text{if } t \geq 1, \end{cases} \tag{9}$$

where $\hat{\mathbf{e}}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|}$. During the base session, we normalize embeddings to treat each sample equally, because otherwise, samples with larger norms would have greater weight in the calculation of $\boldsymbol{w}_j$. As illustrated in Fig. 4(a), there is considerable variance in the norms of samples during the base session. In the incremental sessions, due to the scarcity of samples, normalizing embeddings could lead to a significant loss of information. Therefore, we adopt a staged approach to better accommodate FSCIL.

Norm Distribution. Based on the distribution of class norms and the norms of sample embeddings, we can obtain the norm logits of jth class for ith sample $\mathbf{z}_j^2$ by:

$$\mathbf{z}_j^2 = \begin{cases} p\left(X \geq \ln \|\mathbf{e}_i\| \mid X \sim \mathcal{N}(\mu_j, \sigma_j^2)\right) & \text{if } \ln \|\mathbf{e}_i\| \geq \mu_j \\ p\left(X \leq \ln \|\mathbf{e}_i\| \mid X \sim \mathcal{N}(\mu_j, \sigma_j^2)\right) & \text{if } \ln \|\mathbf{e}_i\| < \mu_j, \end{cases} \tag{10}$$

where X represents a random variable that follows the distribution of norms for class j. We can obtain the probability by the cumulative distribution function of normal distribution. We also explore other distributions in Section 5.4.

Angle-Norm Joint Inference. Since norms contain less information than angles, we first compress the norm logits, ultimately obtaining the joint logits. The final angle-norm joint logits $\mathbf{z}_j$ is calculated by $\mathbf{z}_j = \mathbf{z}_j^1 \left(\mathbf{z}_j^2\right)^C$, where C is the compression coefficient. After compression, the differences between the norm logits of different categories are reduced, thereby amplifying the role of the angle logits. The final predicted category is determined by $\mathbf{z}_j$.

5. Experiments

5.1. Experimental setup

Datasets. Following the previous works setting (Kalla & Biswas, 2022; Song et al., 2023; Tao et al., 2020; Zhou, Wang et al., 2022), we perform conventional experiments on three datasets: CIFAR100 (Krizhevsky et al., 2009), CUB200 (Wah, Branson, Welinder, Perona, & Belongie, 2011) and *mini*ImageNet (Russakovsky et al., 2015). To better reflect real-world situations, we also conduct open-ended and imbalanced experiments on the CompCars (Yang, Luo, Change Loy, & Tang, 2015) dataset. CIFAR100 consists of 100 different classes, with each class containing 600 images. CUB200 contains images of 200 bird species. *mini*ImageNet is a subset of ImageNet (Deng et al., 2009) and consists of 100 selected classes. CompCars is a fine-grained vehicle dataset that contains 431 categories, and the large number of categories allows us to set up long-term learning experiments to simulate open-ended scenarios. In CIFAR100 and *mini*ImageNet, the base session consists of 60 classes, while the incremental sessions comprise 40 classes. During the incremental sessions, the data is organized in a 5-way 5-shot format. As for CUB200, the base session includes 100 classes, and the incremental sessions involve 100 classes as well. In the incremental sessions of CUB200, the data is organized in a 10-way 5-shot format.

Implementation Details. We conduct experiments by separately employing SAAN, and embed it into SOTA methods, *i.e.* FACT (Zhou, Wang et al., 2022), SAVC (Song et al., 2023). Both of these methods adopt the feature freezing paradigm and use NCM classifier. For fairness, all these methods adopt ResNet18 (He et al., 2016) for *mini*ImageNet, CUB200 and CompCars, and ResNet20 (He et al., 2016) for CIFAR100. In SAAN, the initial learning rate is set to 0.1 with cosine annealing on CIFAR100,*mini*ImageNet, and milestone decay on CUB200 and CompCars. The batch size is 256, and training is conducted for 600 epochs. The training parameters in the embedding of FACT and SAVC are set to be the same as those in their works. The selection of hyper-parameters used in SAAN is discussed in Section 5.5.

5.2. Performance on conventional benchmarks

We compare SAAN and methods embedded with SAAN against classical CIL method, *i.e.* , iCaRL (Rebuffi et al., 2017), EEIL (Castro et al., 2018), incremental-trainable FSCIL method, *i.e.* , TOPIC (Tao et al., 2020), frozen paradigm baseline method, Decoupled-Cosine (Vinyals et al., 2016), and incremental-frozen FSCIL methods, *i.e.* , CEC (Zhang et al., 2021), FACT (Zhou, Wang et al., 2022), SAVC (Song et al., 2023), NC-FSCIL (Yang et al., 2023), and M2SD (Lin et al., 2024). Among them, FACT, SAVC, and MS2D are virtual class methods, while NC-FSCIL is a fixed prototype method. Additionally, we present a naive model, referred to as Fine-tune, which refers direct fine-tuning of models using limited data of novel classes. We provide a brief introduction to the comparison methods for further analysis in Appendix B, and show the detailed experiment results in Tables 1–3. Here, the accuracy of the final round and the drop in accuracy from the 0-th round to the final round are used as evaluation metrics.

Compared to Decoupled-Cosine (frozen paradigm baseline), our method exhibits an average improvement over 7.44% across the three datasets in the final session, highlighting the importance of prior spatial allocation. It can be observed that SAAN surpasses FACT on all three datasets, indicating that our method achieves the performance of SOTA models. FACT occupies the sample space with virtual classes, but new classes may not appear within the space occupied by these virtual classes. SAAN further aligns the virtual classes semantically with the new classes, addressing the limitations of FACT, which allows SSAN to achieve better performance.

SAAN does not outperform SAVC, primarily because SAVC not only uses the virtual class method but also incorporates supervised contrastive learning. Based on the self-supervised MoCo (He, Fan, Wu, Xie, & Girshick, 2020) framework, it significantly increases both data volume and training capacity with additional supervisory signals, yielding highly effective results. The generalization capability provided by contrastive learning is crucial for more complex datasets. However, on simpler datasets like CIFAR100, SAAN already outperforms SAVC. This is because SAAN requires generalized features. In simple datasets, the model can easily obtain relatively generalized features, but in more challenging datasets, additional assistance, such as supervised contrastive learning, is needed.

As a plug-in, SAAN is compatible with both FACT and SAVC, demonstrating excellent generalization with virtual class methods. SAVC + SAAN achieves SOTA accuracy in the final round, with improvements of 2.58%, 3.08%, and 3.33% on CIFAR100, CUB200, and *mini*ImageNet, respectively. With the help of SAAN, the average forgetting rates of FACT and SAVC decreased by 2.63% and 2.59%, respectively, across the three datasets. This indicates that allocating space for new classes to reduce the movement of old classes can effectively lower the forgetting rate. SAAN helps SAVC achieve additional improvements, by further refining the spatial positioning of SAVC's semantic virtual classes, and by upgrading the NCM classifier used in SAVC to better leverage the angle and norm information from limited samples. Moreover, the methods combined with SAAN show a relatively smaller decrease in accuracy during continual learning, indicating that SAAN enhances the model's ability for incremental learning, rather than solely improving the recognition of base classes.

SAAN shows significant performance advantages on CUB200 and achieves good results on *mini*ImageNet. However, it performs worse than NC-FSCIL on CIFAR100. One possible explanation is that in SAAN, prototype allocation considers semantic information of the samples, and prototypes are adjustable. This gives SAAN good performance on fine-grained datasets like CUB200, where certain bird species have high similarity, and the similarity between different classes is more important to consider. In contrast, on datasets like CIFAR100, where there is a larger difference between categories, the differences between each category are relatively large, so the importance of carefully adjusting and allocating prototypes may be reduced. Instead, the random allocation and fixed strategy in NC-FSCIL ensure the stability of prototypes during the incremental process, leading to the best performance.

5.3. Comparisons in open-ended and imbalanced scenarios

To better simulate real-world class-incremental task scenarios, we conduct the experiments under open-ended and imbalanced conditions. Unlike the experiments in Section 5.2, we use the CompCars dataset, a fine-grained vehicle dataset with 431 car categories. Among them, 231 classes are used for the base session, while 200 classes are allocated

Table 1
Performance comparison on CUB200 dataset. Drop: The drop in Acc. of the last session relative to session 0.

Method	Acc. in each session (%)↑											Drop↓
	0	1	2	3	4	5	6	7	8	9	10	
Finetune[a]	68.68	43.70	25.05	17.72	18.08	16.95	15.10	10.06	8.93	8.93	8.47	60.21
iCaRL[a] (Rebuffi et al., 2017)	68.68	52.65	48.61	44.16	36.62	29.52	27.83	26.26	24.01	23.89	21.16	47.52
EEIL[a] (Castro, Marín-Jiménez, Guil, Schmid, & Alahari, 2018)	68.68	53.26	47.91	44.20	36.30	27.46	25.93	24.70	23.95	24.13	22.11	46.57
TOPIC (Tao et al., 2020)	68.68	62.49	54.81	49.99	45.25	41.40	38.35	35.36	32.22	28.31	26.28	42.40
Decoupled-Cosine[b] (Vinyals, Blundell, Lillicrap, kavukcuoglu, & Wierstra, 2016)	73.32	68.57	64.37	60.09	58.74	54.76	53.15	50.54	49.48	48.18	46.86	26.46
CEC (Zhang et al., 2021)	75.85	71.94	68.50	62.50	62.43	58.27	57.73	55.81	54.83	53.52	52.28	23.57
NC-FSCIL (Yang et al., 2023)	80.45	75.98	72.30	70.28	68.17	65.16	64.43	63.25	60.66	60.01	59.44	21.01
M2SD (Lin et al., 2024)	81.49	76.67	73.58	68.77	68.73	65.78	64.73	64.03	62.70	62.09	60.96	20.53
SAAN	78.42	74.94	71.54	67.17	66.67	63.38	62.43	61.27	60.49	59.55	58.55	19.87
FACT[b](Zhou, Wang et al., 2022)	77.66	73.70	70.23	65.97	65.24	61.92	61.20	59.51	57.81	57.35	56.11	21.55
FACT+SAAN	78.21	75.12	72.44	67.97	67.19	64.38	63.83	63.29	62.17	61.77	60.22	17.99
SAVC[b](Song et al., 2023)	82.21	78.30	75.25	70.65	70.11	66.96	66.36	65.32	63.61	63.86	62.14	20.07
SAVC+SAAN	**82.55**	**79.81**	**77.04**	**72.39**	**72.42**	**68.97**	**68.83**	**67.62**	**66.25**	**65.94**	**65.22**	**17.33**

[a] FSCIL results implemented by Tao et al. (2020).
[b] Reproduced under the FSCIL setting with the source code.

Table 2
Performance comparison on *mini*ImageNet dataset. Drop: The drop in Acc. of the last session relative to session 0.

Method	Acc. in each session (%)↑									Drop↓
	0	1	2	3	4	5	6	7	8	
Finetune[a]	61.31	27.22	16.37	6.08	2.54	1.56	1.93	2.60	1.40	59.91
iCaRL[a] (Rebuffi et al., 2017)	61.31	46.32	42.94	37.63	30.49	24.00	20.89	18.80	17.21	44.10
EEIL[a] (Castro et al., 2018)	61.31	46.58	44.00	37.29	33.14	27.12	24.10	21.57	19.58	41.73
TOPIC (Tao et al., 2020)	61.31	50.09	45.17	41.16	37.48	35.52	32.19	29.46	24.42	36.89
Decoupled-Cosine[b] (Vinyals et al., 2016)	70.10	64.97	61.01	57.76	54.73	51.93	49.27	47.37	45.77	24.33
CEC (Zhang et al., 2021)	72.00	66.83	62.97	59.43	56.70	53.73	51.19	49.24	47.63	24.37
NC-FSCIL (Yang et al., 2023)	**84.02**	76.80	72.00	67.83	66.35	64.04	61.46	59.54	58.31	25.71
M2SD (Lin et al., 2024)	82.11	**79.92**	**75.44**	**71.31**	68.29	64.32	61.13	58.64	56.51	25.60
SAAN	74.90	69.97	65.59	61.93	58.91	55.82	53.16	50.97	49.98	24.92
FACT[b](Zhou, Wang et al., 2022)	75.65	70.88	66.14	62.33	58.96	55.65	52.73	50.59	48.60	27.05
FACT+SAAN	76.11	71.98	67.21	64.90	60.12	58.83	56.28	54.22	52.11	24.00
SAVC[b](Song et al., 2023)	80.93	75.77	71.96	68.44	65.71	62.22	59.12	57.18	55.98	24.95
SAVC+SAAN	80.97	78.28	74.43	71.03	**68.39**	**65.22**	**62.42**	**60.46**	**59.31**	**21.66**

[a] FSCIL results implemented by Tao et al. (2020).
[b] Reproduced under the FSCIL setting with the source code.

Table 3
Performance comparison on CIFAR100 dataset. Drop: The drop in Acc. of the last session relative to session 0.

Method	Acc. in each session (%)↑									Drop↓
	0	1	2	3	4	5	6	7	8	
Finetune[a]	64.10	39.61	15.37	9.80	6.67	3.80	3.70	3.14	2.65	61.45
iCaRL[a] (Rebuffi et al., 2017)	64.10	53.28	41.69	34.13	27.93	25.06	20.41	15.48	13.73	50.37
EEIL[a] (Castro et al., 2018)	64.10	53.11	43.71	35.15	28.96	24.98	21.01	17.26	15.85	48.25
TOPIC (Tao et al., 2020)	64.10	55.88	47.07	45.16	40.11	36.38	33.96	31.55	29.37	34.73
Decoupled-Cosine[b] (Vinyals et al., 2016)	74.02	68.71	64.24	60.25	56.91	53.87	51.57	49.57	47.54	26.48
CEC (Zhang et al., 2021)	73.07	68.88	65.26	61.19	58.09	55.57	53.22	51.34	49.14	**23.93**
NC-FSCIL (Yang et al., 2023)	**82.52**	**76.82**	**73.34**	**69.68**	**66.19**	**62.85**	**60.96**	**59.02**	**56.11**	26.41
SAAN	78.58	73.02	68.99	65.19	61.90	59.82	56.99	54.74	53.98	24.60
FACT[b](Zhou, Wang et al., 2022)	79.20	73.05	69.04	64.89	61.44	58.82	56.48	54.20	51.93	27.27
FACT+SAAN	78.11	72.98	69.21	64.90	62.12	59.83	57.28	54.22	52.11	26.00
SAVC[b](Song et al., 2023)	78.60	73.42	68.89	64.63	61.34	58.32	56.14	53.85	51.64	26.96
SAVC+SAAN	79.43	74.02	69.90	65.89	62.90	60.18	58.27	56.41	54.22	25.21

[a] FSCIL results implemented by Tao et al. (2020).
[b] Reproduced under the FSCIL setting with the source code.

for the incremental sessions. To simulate long-term few-shot class-incremental tasks, we extend the number of sessions to 21, including 20 incremental sessions. Additionally, the number of categories in each incremental session is sampled from a Gaussian distribution with a mean of 10 and a variance of 5, rather than being fixed at 10 classes. Similarly, the number of samples per category is sampled from a Gaussian distribution with a mean of 5 and a variance of 2, rather than being fixed at 5 samples. This setup is designed to simulate the class and sample imbalance characteristics in the real world. Due to the randomness in the task setup, we conduct 3 repeated experiments based on this configuration, and all experimental data are presented as the average of repeated experiments.

Table 4
The experimental results under open-ended and imbalanced conditions on CompCars dataset. LA: the Last session Accuracy (%), AA: the Average Accuracy (%), Δ_{last}: Relative improvements of the last sessions compared to the Decoupled-Cosine, $\Delta_{average}$: Relative improvements of the last sessions compared to the Decoupled-Cosine, Drop: The drop in accuracy of the last session relative to session 0.

Method	LA ↑	Δ_{last}	AA ↑	$\Delta_{average}$	Drop ↓
Decoupled-Cosine (Vinyals et al., 2016)	56.24	–	69.23	–	29.76
SAAN	62.27	+6.03	76.24	+7.01	31.38
FACT (Zhou, Wang et al., 2022)	61.55	+5.31	75.23	+6.00	31.23
FACT+SAAN	63.59	+7.35	76.52	+7.29	29.60
SAVC (Song et al., 2023)	70.76	+14.52	81.32	+12.09	24.24
SAVC+SAAN	**73.59**	**+17.35**	**83.33**	**+14.10**	**21.93**

We perform experiments on Decoupled-Cosine, FACT, SAVC and SAAN, and integrate our method as a plug-in into FACT and SAVC, like Section 5.2. To ensure fairness, all methods are implemented using the same random seeds. We show the experiment results in Table 4, and the detailed results are provided in Appendix C. Here, the Last session Accuracy (LA), the Average Accuracy of all sessions (AA) and The drop in accuracy of the last session relative to session 0 (Drop) are used as evaluation metrics.

The experimental results are largely consistent with those in Section 5.2, particularly resembling the outcomes on CUB200 in Table 1. This aligns with our expectations, as both being fine-grained datasets, CompCars poses similar challenges to these methods as CUB200 does. SAAN shows an improvement of 6.03% in LA and 7.01% in AA over decoupled-cosine (the frozen paradigm baseline), and an enhancement of 0.72% in LA and 1.01% in AA over FACT, indicating that our method can achieve the performance of SOTA models when used independently, and remains effective in open-ended and imbalanced scenarios. SAAN does not outperform SAVC, which is also in line with our previous analysis. Compared to coarse-grained datasets with distinct inter-class differences such as CIFAR100, models require greater generalization capability on fine-grained datasets, and the supervised contrastive learning used in SAVC can well compensate for this. Therefore, when we integrate SAAN as a plugin into SAVC, the model's performance significantly surpasses that of either SAAN or SAVC alone, with a 2.83% improvement in LA and a 2.01% improvement in AA over SAVC, achieving the highest performance and the lowest forgetting rate. SAAN enhances the performance of SAVC through two key refinements: it meticulously adjusts the spatial arrangement of SAVC's semantic virtual categories and enhances the NCM classifier within SAVC to more effectively utilize the angular and magnitude data derived from a constrained set of samples. Embedding SAAN into FACT also boosts FACT's performance. This demonstrates that SAAN remains effective as a plugin in open-ended and imbalanced scenarios.

5.4. Ablation studies

We conduct ablation experiments to verify the importance of the components of our method. Building upon the incremental freezing framework, we set Decoupled-Cosine as the baseline and then gradually add $\mathcal{L}_1$, $\mathcal{L}_2$, 2SNCM (**2S**), and Norm Distribution (**ND**) to observe their respective contributions. The results on the CIFAR100 dataset are presented in Table 5.

The experiments reveal that $\mathcal{L}_1$ has the most significant impact, with a 3.41% improvement when combined with the baseline, highlighting the importance of embedding space allocation. $\mathcal{L}_2$ aims to increase inter-class distances, resulting in a 4.01% improvement. 2SNCM achieves a modest improvement of 4.19%, indicating a limited enhancement. This indicates that 2SNCM can achieve more reasonable representative points simply by normalizing and averaging the samples in the base round, ultimately leading to a certain performance improvement. To further verify the effectiveness of 2SNCM, we present multiple repeated experiments on three datasets in Appendix E. Norm Distribution achieves an improvement of 5.18%, demonstrating the effectiveness of introducing norm information to the classifier. The ANJ, composed of 2SNCM and Norm Distribution, ultimately achieves a performance improvement of 5.44%. The components in ANJ do not participate in model training but construct a more accurate classifier based on the feature extractor, with minimal computational cost. In summary, the spatial allocation strategy of CCSA is highly beneficial for FSCIL, while ANJ aids classification by making more comprehensive and effective use of the features.

To verify the rationality of the distribution assumption in ANJ and the effectiveness of the shared distribution technique, we conduct an ablation study on ANJ alone, as shown in Table 6. The log-normal, normal, and Weibull distributions can effectively represent feature norms, leading to a certain improvement in model performance. However, the Pareto distribution, a heavy-tailed distribution, rendered ANJ ineffective due to its significant difference from the norm distribution. Among all distributions, the Weibull and log-normal distributions performed the best. The log-normal distribution has the advantage of simple parameter estimation, whereas the Weibull distribution requires storing data for distribution fitting. Without adopting the shared distribution technique, the incremental data becomes too sparse, leading to a significant discrepancy between the estimated distribution and the true distribution. This issue is particularly severe for the Weibull distribution, which requires a sufficient amount of data for proper fitting. Regardless of the chosen distribution, without the shared distribution technique, the performance deteriorates significantly. By leveraging the shared distribution technique, the model can better identify the session to which a sample belongs, thereby indirectly improving the final classification performance.

5.5. Numerical analysis

Visualization of Embedding Separation. We randomly choose 6 classes from base classes and 3 classes from novel classes respectively and visualize the embeddings of them on the CIFAR100 dataset with t-SNE (Van der Maaten & Hinton, 2008) in Fig. 5. Firstly, it can be observed that the relative positions of class embeddings shift compared to Decoupled-Cosine. In Decoupled-Cosine, the purple, brown, and red classes are adjacent, while in SAAN, the green, black, and red classes are adjacent. This is because the allocation of class centers affects the positions of class means. During the class increment process, in SAAN, the purple and brown new classes are assigned to two different reserved spaces, whereas in Decoupled-Cosine, the purple and brown classes are assigned to adjacent spaces and are very close to each other. This makes it difficult to distinguish between the purple and brown classes in Decoupled-Cosine. While in SAAN, it is easier to identify them. Additionally, in SAAN, the inter-class distance is larger, while the intra-class distance is smaller.

In Fig. 5(b), it can be observed that the sample embeddings are distributed around the centers we set, indicating that the limiting effect of the centers is effective. There is some deviation between these centers and the class mean, which is due to the influence of cross-entropy.

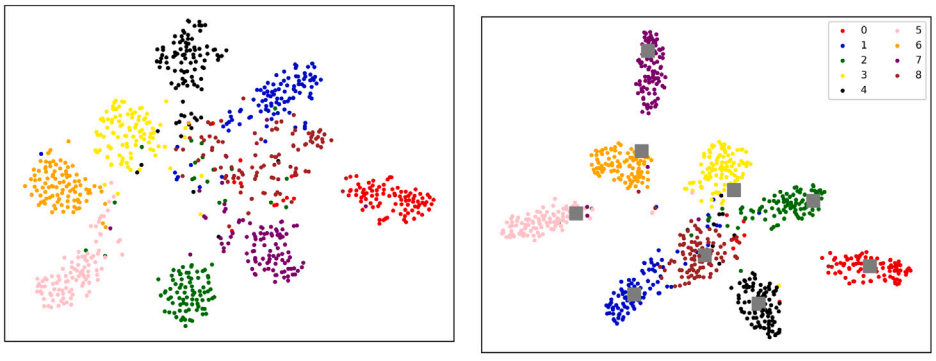
Performance on Novel and Base Classes. Solely based on the accuracy over all classes, we cannot evaluate how much new knowledge the model has learned and how much old knowledge it has forgotten. We report the accuracy of the base classes and the novel classes as well as harmonic mean in Fig. 6. With the help of SAAN, SAVC's accuracy for novel classes improves by 7.20% and 3.76% in CIFAR100 and CUB200 respectively. The accuracy improvement for old classes is very limited compared to that for new classes. SAAN obviously enhances the compatibility of the model to new classes verifying that limiting the space occupation of old classes and allocating space for new classes are conducive to the learning of new classes.

Table 5
Ablation studies on CIFAR100 dataset. Δ_{last}: Relative improvements of the last sessions compared to the baseline.

$\mathcal{L}_1$	$\mathcal{L}_2$	2S	ND	Acc. in each session (%)↑									Δ_{last}
				0	1	2	3	4	5	6	7	8	
				74.02	68.71	64.24	60.25	56.91	53.87	51.57	49.57	47.54	–
✓				77.05	71.66	67.27	63.56	60.30	57.55	55.12	53.08	50.95	+3.41
✓	✓			77.63	72.23	68.22	64.28	60.94	58.12	55.70	53.58	51.55	+4.01
✓	✓	✓		78.08	72.45	68.42	64.45	61.18	58.39	55.91	53.79	51.73	+4.19
✓	✓		✓	78.10	72.79	68.73	65.05	61.63	59.63	56.74	54.58	52.72	+5.18
✓	✓	✓	✓	78.58	73.02	68.99	65.19	61.90	59.82	56.99	54.74	52.98	+5.44

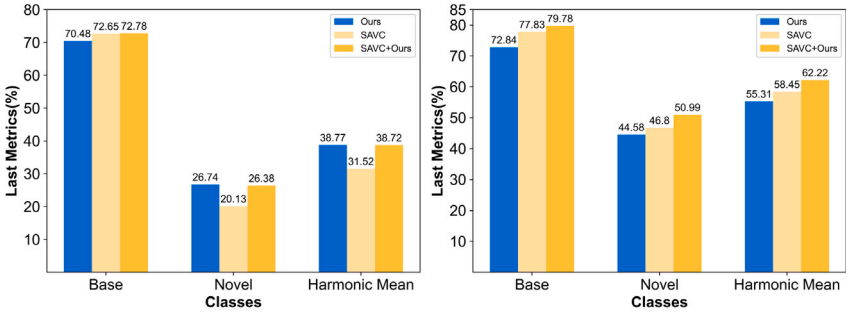
Table 6
ANJ ablation study on CUB200 dataset.

Distribution	Shared distribution	Acc. in each session (%)↑										
		0	1	2	3	4	5	6	7	8	9	10
Normal	×	78.19	73.84	70.29	66.36	65.68	62.11	61.55	60.10	59.69	58.20	57.56
	✓	78.19	74.56	71.14	66.79	66.29	63.02	62.03	60.87	60.01	59.15	58.18
Log-Normal	×	78.42	74.03	70.63	66.39	65.90	62.52	61.82	60.44	59.66	58.72	57.72
	✓	78.42	74.94	71.54	67.17	66.67	63.38	62.43	61.27	60.49	59.55	58.55
Weibull	×	78.43	74.48	70.90	66.75	65.94	62.80	61.63	60.18	59.68	58.96	57.70
	✓	78.43	74.98	71.58	67.30	66.62	63.48	62.47	61.16	60.51	59.71	58.65
Patero	×	76.44	72.03	68.33	64.30	63.85	60.09	60.02	58.62	57.61	56.86	55.46
	✓	76.44	72.30	68.62	64.70	64.14	60.45	60.48	58.95	57.99	57.19	55.76



(a) Decoupled-Cosine. (b) SAAN.

Fig. 5. The t-SNE visualization on CIFAR100 dataset of the embeddings. Classes 0–5 represent the base classes while classes 6–8 represent the novel classes. The squares represent the centers set by SAAN.



(a) CIFAR100. (b) CUB200.

Fig. 6. Accuracy of base and novel classes at the last session.

Parameter Sensitivity. We conduct a search and discussion on the loss weights α, β, compression coefficient C, as well as the center moving rate η and its decay rate λ. The grid search results for cosine center loss weights and center moving rate with its decay rate on CUB200 are shown in Fig. 7. The parameter analysis across three datasets is presented in Fig. 8. As α and β increase, the effect of CCSA becomes stronger, leading to improved model performance, which reaches its optimum when α is 2 and β is 0.4. $\mathcal{L}_1$ serves to push away from other centers, while $\mathcal{L}_2$ pulls closer to the corresponding center. If β is too large and α is too small, the pushing effect will be dominant,

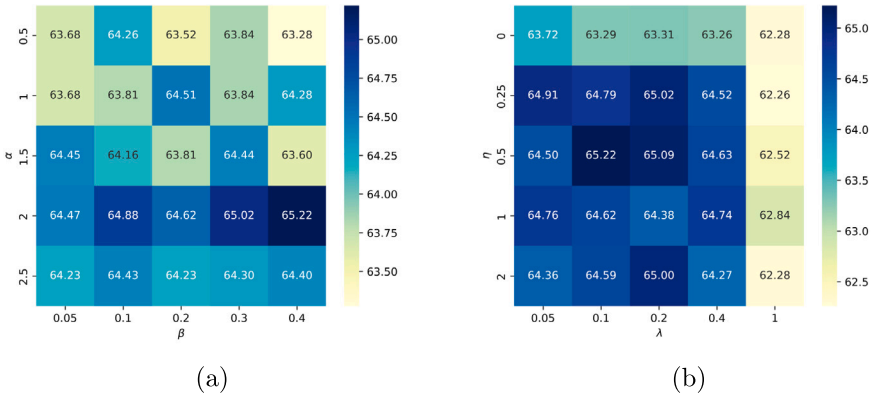


Fig. 7. The parameter grid search for the last session accuracy of SAVC + SAAN on the CUB200 dataset. (a) The weights of $\mathcal{L}_1$ and $\mathcal{L}_2$. (b) The center moving rate and its decay rate.

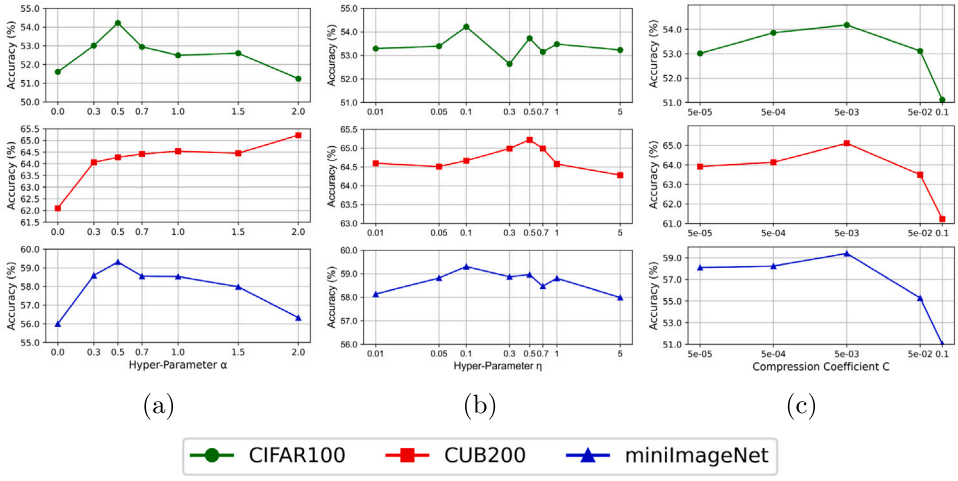


Fig. 8. Last session accuracy of SAVC + SAAN with different hyper-parameters on three datasets. (a) α varies while η is fixed and the ratio of α to β is 5. (b) η varies while α is fixed, and λ is 0.1. (c) C varies.

while the more crucial pulling effect will be weakened, leading to a negative impact. For all three datasets, the optimal value of C is consistently 0.005. When C exceeds 0.05, the influence of norm logits becomes overly dominant, rendering the more important angular logits ineffective, which leads to a sharp performance drop.

It can be observed that the center moving rate and decay rate are relatively insensitive, as values ranging from 0.25 to 2 all have a positive effect. However, considering two special cases: when the center moving rate is 0, it represents fixed centers; when the center moving decay rate is 1, it means that the centers are continuously updated without playing a role in spatial allocation, only contributing to intra-class cohesion. Fixed centers perform better than the baseline but are inferior to the learnable SAAN. This indicates that spatial allocation is beneficial for FSCIL, and learnable centers achieve better allocation performance. Continuously updating centers perform almost the same as the baseline, failing to achieve spatial allocation. This highlights the importance of spatial allocation, as merely increasing inter-class distance has limited effectiveness.

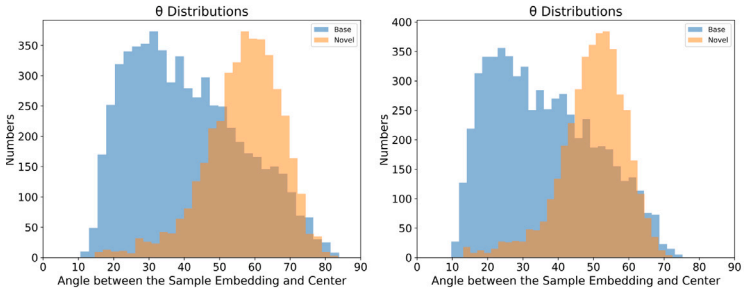
Angle Distribution from the Centers. We present the angle of each sample relative to its center of SAVC and SAVC+SAAN, as shown in Fig. 9. It can be observed that the mean angle for novel classes of SAVC is approximately 60 degrees, while the mean angle of SAVC+SAAN is around 50 degrees, with a noticeably smaller maximum angle compared to SAVC. The results on base classes are similar. This implies that SAVC+SAAN exhibits a more compact intra-class structure, by using our cosine center loss.

Normal Test.

We present the Quantile–Quantile plot of the log norms of sample embeddings in Fig. 10, where the horizontal axis represents theoretical quantities and the vertical axis represents sample quantities. Due to the granularity of the base session distribution being categories, we randomly select one category. During the incremental sessions, we select one session due to distribution sharing. It can be observed that the data points are nearly aligned along a straight line, indicating that the sample log norms can be considered to follow a normal distribution. In session 8, the data points slightly deviate above the straight line, indicating that the shared norm distribution exhibits a long-tail behavior.

6. Conclusion

We summarize two challenges present in the FSCIL problem: (i) Due to the CIL problem setup, current classes occupy the entire feature space, which is detrimental to learning new classes. (ii) Due to



(a) SAVC (b) SAVC+SAAN

Fig. 9. The angle distributions between the sample embeddings and their corresponding class centers on *mini*ImageNet dataset. Blue represents base classes, and orange represents novel classes.

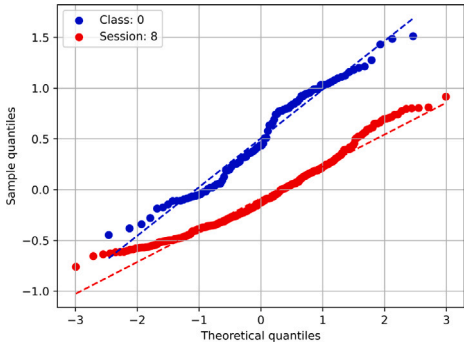


Fig. 10. Quantile–Quantile plot of embedding log norms on *mini*ImageNet dataset.

the FSL problem setup, the small number of samples in incremental rounds is insufficient for fully training. To address these two issues and improve upon previous methods, we propose the SAAN framework. In response to challenge (i), SSAN sets the orthogonal class centers to partition the embedding space, and guides sample embeddings toward their corresponding class center, thus providing reserved space for new classes. Experiments show that under the effect of spatial allocation, the model performance improves significantly, highlighting the importance of spatial allocation. Additionally, the spatial allocation and virtual class reservation methods are compatible and jointly reserve space for new classes. In response to challenge (ii), SAAN establishes a more precise decision boundary by integrating 2SNCM and ND, making full use of angle and norm information. Experiments show that reasonably selecting representative points for classes and fully utilizing the complete embedding information to optimize the classifier, without changing the feature extraction module, still contributes to performance improvement.

CRediT authorship contribution statement

Dunwei Tu: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Huiyu Yi:** Writing – review & editing, Writing – original draft, Visualization, Validation, Formal analysis, Conceptualization. **Tieyi Zhang:** Writing – review & editing, Writing – original draft, Visualization, Validation, Formal analysis, Data curation. **Ruotong Li:** Writing – review & editing, Visualization, Validation, Data curation. **Furao Shen:** Writing – review & editing, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Jian Zhao:** Writing – review & editing, Supervision, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the STI 2030-Major Projects of China under Grant 2021ZD0201300, and the Fundamental Research Funds for the Central Universities, China under Grant 2024300394.

Appendix A. Gradient analysis of cosine center loss

In this section, we provide the gradient of the proposed cosine center loss and analyze the update mechanism of the learnable centers. Since analyzing the gradients of $\mathcal{L}_1$ and $\mathcal{L}_2$ with respect to embeddings requires deriving the gradient of the cosine similarity with respect to the vector, we give it in advance:

$$\frac{\partial \cos\langle \mathbf{a}, \mathbf{b}\rangle}{\partial \mathbf{a}} = \frac{\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|} - \frac{\mathbf{a}}{\|\mathbf{a}\|^2}\cos\langle \mathbf{a}, \mathbf{b}\rangle, \tag{A.1}$$

where $\mathbf{a}$ and $\mathbf{b}$ are any vectors. In this paper, the norm specifically refers to the L2 norm.

A.1. Analysis of $\mathcal{L}_1$

First we give the negative gradient of $\mathcal{L}_1$ with respect to $\hat{\mathbf{e}}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|}$ as:

$$-\frac{\partial \mathcal{L}_1}{\partial \hat{\mathbf{e}}_i} = \frac{1}{m}\hat{\mathbf{c}}_{y_i}. \tag{A.2}$$

Since $\mathbf{c}_{y_i}$ is normalized upon initialization and after each updating, $\hat{\mathbf{c}}_{y_i}$ is equal to $\mathbf{c}_{y_i}$. Eq. (A.2) shows that the normalized sample embedding will move closer to the center according to vector addition. Combining Eq. (A.1) or using chain derivation role, we can further obtain the negative gradient of $\mathcal{L}_1$ with respect to $\mathbf{e}_i$ as:

$$\begin{aligned} -\frac{\partial \mathcal{L}_1}{\partial \mathbf{e}_i} &= -\frac{\partial \mathcal{L}_1}{\partial \hat{\mathbf{e}}_i}\frac{\partial \hat{\mathbf{e}}_i}{\partial \mathbf{e}_i} \\ &= \frac{1}{m}\left(\frac{\mathbf{c}_{y_i}}{\|\mathbf{e}_i\|} - \frac{\mathbf{e}_i \cos\langle \mathbf{e}_i, \mathbf{c}_{y_i}\rangle}{\|\mathbf{e}_i\|^2}\right). \end{aligned} \tag{A.3}$$

Eq. (A.3) indicates that the sample embedding will move perpendicular to itself, pointing toward the center. $\mathcal{L}_1$ guides the samples, playing a role in spatial allocation and reducing the intra-class distance. $\mathcal{L}_2$ pushes the samples away from other centers, helping to increase the inter-class distance.

A.2. Analysis of $\mathcal{L}_2$

Likewise, we give the gradient of $\mathcal{L}_2$ with respect to $\hat{\mathbf{e}}_i$ as:

$$\frac{\partial \mathcal{L}_2}{\partial \hat{\mathbf{e}}_i} = \frac{1}{m}\frac{1}{|\mathcal{Y}|}\sum_{j=1}^{|\mathcal{Y}|}\mathbf{c}_j \mathbb{I}\left(y_i \neq j\right). \tag{A.4}$$

Eq. (A.4) shows that the normalized embedding will move away from the mean direction of other class centers through vector subtraction. Similarly, combining Eq. (A.1) or using chain derivation role, we further obtain the gradient of $\mathcal{L}_2$ with respect to $\mathbf{e}_i$ as:

$$\begin{aligned}\frac{\partial \mathcal{L}_2}{\partial \mathbf{e}_i} &= \frac{\partial \mathcal{L}_2}{\partial \hat{\mathbf{e}}_i}\frac{\partial \hat{\mathbf{e}}_i}{\partial \mathbf{e}_i} \\ &= \frac{1}{m}\frac{1}{|\mathcal{Y}|}\sum_{j=1}^{|\mathcal{Y}|}\left(\frac{\mathbf{c}_j}{\|\mathbf{e}_i\|} - \frac{\mathbf{e}_i \cos\langle \mathbf{e}_i, \mathbf{c}_j\rangle}{\|\mathbf{e}_i\|^2}\right)\mathbb{I}\left(y_i \neq j\right).\end{aligned} \tag{A.5}$$

Eq. (A.5) means that the sample embedding will move perpendicular to itself and point in the opposite direction to the mean of other centers. The gradient direction is the fastest direction in which the cosine similarity between the embedding and centers of other classes decreases.

A.3. The center update mechanism

Another way to update the centers is to set them as learnable parameters and update them through gradient descent based on $\mathcal{L}_1$. Since we are concerned with the normalized center, we compute the gradient of $\mathcal{L}_1$ with respect to the normalized center as:

$$\frac{\partial \mathcal{L}_1}{\partial \hat{\mathbf{c}}_j} = -\frac{1}{m}\sum_{i=1}^{m}\left(\hat{\mathbf{e}}_i \mathbb{I}(y_i = j)\right). \tag{A.6}$$

Update the center through gradient descent as:

$$\mathbf{c}_j = \hat{\mathbf{c}}_j - \eta\frac{\partial \mathcal{L}_1}{\partial \hat{\mathbf{c}}_j}. \tag{A.7}$$

It can be observed that gradient descent updates are equivalent to updates via vector addition. That is, the centers can be adjusted to positions more aligned with the samples either through gradient descent or vector addition, reducing the unreasonable allocation from pre-assigned centers.

Appendix B. Comparison methods

To analyze the performance comparison with other methods, we provide an introduction to the comparison methods here.

- **Finetune**. It directly fine-tunes the model with limited data in incremental sessions, without addressing either the forgetting problem or overfitting.
- **iCaRL** (Rebuffi et al., 2017). It maintains an "episodic memory" of the exemplars, which enables one to incrementally learn new classes without forgetting old classes.
- **EEIL** (Castro et al., 2018). It evolves ensemble models gradually, adapting to new data while preserving past knowledge.
- **TOPIC** (Tao et al., 2020). It mitigates forgetting by utilizing a Neural Gas (NG) network (Martinetz, Schulten, et al., 1991) to preserve the topology of the feature manifold formed by different classes.
- **Decoupled-Cosine** (Vinyals et al., 2016). It is a one-shot learning method that utilizes cosine similarity to efficiently learn new classes with minimal data by decoupling the classifier from the feature extractor.
- **CEC** (Zhang et al., 2021). It introduces an auxiliary graph model that promotes the exchange of contextual information among classifiers, thereby improving the adaptation process.
- **FACT** (Zhou, Wang et al., 2022). It pre-assigns multiple virtual prototypes and generates virtual instances via instance mixture in the embedding space, to reserve spaces for incoming new classes.
- **NC-FSCIL** (Yang et al., 2023). It introduces a neural collapse-inspired feature-classifier alignment method for FSCIL, addressing the misalignment issue between features and classifiers in incremental learning with superior performance demonstrated on multiple datasets.
- **M2SD** (Lin et al., 2024). It proposes a dual-branch structure that facilitates the expansion of the feature space and employs self-distillation to pass additional enhanced information back to the base network.
- **SAVC** (Song et al., 2023). It introduces more semantic knowledge by imagining virtual classes which not only hold enough feature space for future updates but enable a multi-semantic aggregated inference effect.

Appendix C. Detailed results of the open-ended experiments

To better simulate the real-world environment of class-incremental tasks, we conduct this comparative experiment under open-ended and imbalanced conditions. The experimental setup and the analysis are elaborated in Section 5.3. The experimental results we obtain are presented in Table C.7.

Appendix D. Computational cost analysis

We analyze the computational cost of SAAN compared to other methods on CUB200 dataset. The additional cost introduced by SAAN mainly comes from the increased parameters of trainable centers and the inference computation brought by ANJ. Therefore, we report the extra training parameters and inference computation introduced by SAAN, as well as the parameter count and inference computation of other methods themselves, as shown in Table D.8. It can be observed that the additional parameter count and inference computation introduced by SAAN are both less than 1%. In SAVC, the computational cost and center parameter count are both twice as high as in other methods because SAVC performs category augmentation with virtual classes, leading to a doubled number of centers. During inference, both the original and augmented samples need to be processed, resulting in increased inference computation.

Appendix E. Repeated experiments on 2SNCM

2SNCM and NCM only differ in the selection of representative points. In order to exclude random perturbations, we conducted multiple repeated experiments on them. Here the seeds are canceled and multiple repeated experiments are conducted. We use Decoupled-Cosine to conduct 5 replicate experiments on 3 datasets respectively and report the average performance of them in Table E.9. Compared with NCM, 2SNCM improves the average accuracies on CIFAR100, CUB200 by 0.16%. On *mini*ImageNet, 2SNCM performs better in the early incremental phase and base phase, and NCM performs better in the late incremental phase. The better performance of 2SNCM indicates the effectiveness of normalizing based on angles when there are enough base round samples while preserving norm information without normalization when the incremental round samples are fewer.

Data availability

Data will be made available on request.

Table C.7
Detailed results of the open-ended and imbalanced experiments on CompCars dataset.

Method	Random	Acc. in each session (%)↑																			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Decoupled-Cosine (Vinyals et al., 2016)	rand 1	85.88	83.90	82.88	81.79	80.57	79.13	76.44	73.40	71.61	69.15	67.27	64.40	63.34	61.57	60.58	59.77	58.71	57.87	57.44	56.30	56.22
	rand 2	86.26	85.02	83.89	81.68	80.69	79.15	76.53	74.11	71.93	69.49	67.27	64.80	63.56	61.95	60.94	59.02	58.58	57.72	56.96	56.37	55.56
	rand 3	85.88	84.36	83.16	82.21	80.85	79.46	77.00	74.30	72.37	70.07	67.90	65.29	64.10	62.32	61.30	60.50	59.52	58.53	58.10	57.38	56.96
SAAN	rand 1	93.79	91.89	90.70	89.80	88.51	86.98	84.44	81.53	79.79	77.41	75.16	72.23	71.24	69.12	68.15	67.06	66.05	64.96	64.33	63.37	62.91
	rand 2	93.40	91.80	90.45	88.83	87.77	85.96	83.08	80.40	78.36	75.57	73.44	70.78	69.55	67.75	66.56	64.62	64.09	63.15	62.29	61.56	60.82
	rand 3	93.79	91.95	90.97	90.07	88.78	87.31	84.83	81.97	80.26	77.58	75.49	72.58	71.35	69.28	68.09	67.05	66.06	65.01	64.39	63.50	63.09
FACT (Zhou, Wang et al., 2022)	rand 1	92.76	91.00	89.61	88.74	87.18	85.73	83.10	80.07	78.25	75.64	73.23	71.03	69.55	67.72	66.71	65.74	64.75	63.80	63.30	62.48	62.10
	rand 2	92.82	91.40	89.59	87.96	86.97	85.24	82.64	79.98	77.93	75.18	72.91	70.31	68.96	67.08	66.00	63.97	63.54	62.67	61.83	61.16	60.15
	rand 3	92.76	90.85	89.81	88.87	87.43	85.93	83.31	80.44	78.79	76.17	73.76	71.56	70.06	68.07	67.00	66.06	65.13	64.05	63.54	62.83	62.40
FACT+SAAN	rand 1	93.44	91.44	89.77	88.92	87.87	86.27	84.05	81.41	79.64	77.16	75.21	72.90	71.48	69.49	68.49	67.75	66.92	66.00	65.54	64.78	64.40
	rand 2	92.54	90.72	89.53	88.05	87.07	85.26	82.76	80.20	78.23	75.48	73.70	71.06	69.83	68.02	66.92	64.96	64.55	63.62	62.76	62.26	61.46
	rand 3	93.61	92.19	90.65	89.85	88.70	87.05	84.89	82.48	80.70	78.43	76.02	73.74	72.32	70.32	69.37	68.32	67.50	66.49	65.97	65.56	64.92
SAVC (Song et al., 2023)	rand 1	95.31	93.18	91.92	91.10	90.53	89.01	87.43	85.07	84.05	81.91	80.29	78.37	77.59	76.22	75.47	74.98	74.20	73.35	73.11	72.09	71.85
	rand 2	94.38	92.37	91.45	90.39	89.11	87.78	86.11	84.20	82.53	80.34	78.65	76.71	75.38	73.93	73.11	71.36	71.05	70.32	69.44	68.82	68.10
	rand 3	95.31	93.69	92.42	91.75	90.97	89.77	88.16	86.08	85.06	82.98	81.54	79.36	78.33	76.81	76.11	75.36	74.51	73.84	73.36	72.93	72.33
SAVC+SAAN	rand 1	95.60	94.13	92.86	92.52	91.74	90.40	88.73	86.63	85.77	83.91	83.02	80.95	79.94	78.59	77.74	77.36	76.63	75.71	75.45	74.56	74.37
	rand 2	95.35	94.08	93.52	92.25	91.07	89.83	87.96	86.36	84.87	82.71	81.30	79.81	78.89	77.37	76.63	75.10	74.58	73.78	72.88	72.37	71.86
	rand 3	95.60	94.63	93.46	92.79	91.91	90.68	89.18	87.32	86.23	84.37	83.02	81.12	80.18	78.74	77.78	77.20	76.46	75.67	75.27	74.57	74.53

Table D.8
Analysis of trainable parameter count and inference computational cost for different methods on the CUB200 dataset. The computational cost is measured with a batch size of 100.

Method	Model parameters (M)	Inference cost (GFLOPs)	Center parameters (M)	ANJ cost (GFLOPs)
Decoupled-Cosine (Vinyals et al., 2016)	11.18	182.35	+0.0512	+1.68
FACT (Zhou, Wang et al., 2022)	11.18	182.35	+0.0512	+1.68
SAVC (Song et al., 2023)	11.50	364.77	+0.1024	+3.27

Table E.9
Comparison 2SNCM with NCM on three mainstream datasets in FSCIL. Δ_{avg}: The average of the accuracies on all sessions. The data comes from the average accuracy of 5 random experiments.

Dataset	Method	Acc. in each session (%)↑											Δ_{avg}
		0	1	2	3	4	5	6	7	8	9	10	
CIFAR100	NCM	74.17	68.80	64.33	60.37	57.06	53.92	51.71	49.68	47.66	–	–	58.63
CIFAR100	2SNCM	74.35	69.03	64.56	60.60	57.30	54.11	51.81	49.65	47.70	–	–	58.79
CUB200	NCM	73.18	68.41	64.32	60.05	58.67	54.71	53.02	50.42	49.38	48.09	46.79	57.00
CUB200	2SNCM	73.28	68.58	64.51	60.24	58.86	54.88	53.19	50.58	49.52	48.17	46.93	57.16
*mini*ImageNet	NCM	70.32	65.29	61.25	58.01	55.09	52.11	49.51	47.66	46.01	–	–	56.14
*mini*ImageNet	2SNCM	70.46	65.33	61.27	58.01	55.04	52.04	49.42	47.59	45.91	–	–	56.12

References

Ahmed, N., Kukleva, A., & Schiele, B. (2024). OrCo: Towards better generalization via orthogonality and contrast for few-shot class-incremental learning. In *CVPR* (pp. 28762–28771).

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., & Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. In *ECCV* (pp. 139–154).

Arnold, S., Iqbal, S., & Sha, F. (2021). When maml can adapt fast and how to assist when it cannot. In *AISTATS* (pp. 244–252). PMLR.

Castro, F. M., Marín-Jiménez, M. J., Guil, N., Schmid, C., & Alahari, K. (2018). End-to-end incremental learning. In *ECCV* (pp. 233–248).

Cheraghian, A., Rahman, S., Fang, P., Roy, S. K., Petersson, L., & Harandi, M. (2021). Semantic-aware knowledge distillation for few-shot class-incremental learning. In *CVPR* (pp. 2534–2543).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *CVPR* (pp. 248–255). http://dx.doi.org/10.1109/CVPR.2009.5206848.

Dong, S., Hong, X., Tao, X., Chang, X., Wei, X., & Gong, Y. (2021). Few-shot class-incremental learning via relation knowledge distillation. *35*, In *AAAI* (pp. 1255–1263).

Fu, Y., Fu, Y., & Jiang, Y.-G. (2021). Meta-fdmixup: Cross-domain few-shot learning guided by labeled target data. In *MM* (pp. 5326–5334).

Goldberg, Y., & Levy, O. (2014). word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722.

Guerriero, S., Caputo, B., & Mensink, T. (2018). DeepNCM: Deep nearest class mean classifiers.

Han, X., Papyan, V., & Donoho, D. L. (2021). Neural collapse under mse loss: Proximity to and dynamics on the central path. arXiv preprint arXiv:2106.02073.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *CVPR* (pp. 9729–9738).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR* (pp. 770–778).

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Kalla, J., & Biswas, S. (2022). S3C: Self-supervised stochastic classifiers for few-shot class-incremental learning. In *ECCV* (pp. 432–448). Springer.

Kothapalli, V. (2022). Neural collapse: A review on modelling principles and generalization. arXiv preprint arXiv:2206.04041.

Krizhevsky, A., Hinton, G., et al. (2009). *Learning multiple layers of features from tiny images*. Toronto, ON, Canada.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1–2), 83–97.

Li, Z., & Hoiem, D. (2017). Learning without forgetting. *TPAMI*, *40*(12), 2935–2947.

Lin, J., Wu, Z., Lin, W., Huang, J., & Luo, R. (2024). M2SD: Multiple mixing self-distillation for few-shot class-incremental learning. *38*, In *AAAI* (pp. 3422–3431).

Liu, B., Cao, Y., Lin, Y., Li, Q., Zhang, Z., Long, M., et al. (2020). Negative margin matters: Understanding margin in few-shot classification. In *ECCV* (pp. 438–455). Springer.

Liu, X., Masana, M., Herranz, L., Van de Weijer, J., Lopez, A. M., & Bagdanov, A. D. (2018). Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *ICPR* (pp. 2262–2268). IEEE.

Martinetz, T., Schulten, K., et al. (1991). *A" neural-gas" network learns topologies*. University of Illinois at Urbana-Champaign Champaign, IL.

Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., & Van De Weijer, J. (2022). Class-incremental learning: survey and performance evaluation on image classification. *TPAMI*, *45*(5), 5513–5533.

Mensink, T., Verbeek, J., Perronnin, F., & Csurka, G. (2013). Distance-based image classification: Generalizing to new classes at near-zero cost. *TPAMI*, *35*(11), 2624–2637.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *26*, In *NIPS*.

Mittal, S., Galesso, S., & Brox, T. (2021). Essentials for class incremental learning. In *CVPR* (pp. 3513–3522).

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, *5*(1), 32–38.

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999.

Oreshkin, B., Rodríguez López, P., & Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. *31*, In *NeurIPS*.

Rajasegaran, J., Khan, S., Hayat, M., Khan, F. S., & Shah, M. (2020). Self-supervised knowledge distillation for few-shot learning. arXiv preprint arXiv:2006.09785.

Ravi, S., & Larochelle, H. (2016). Optimization as a model for few-shot learning. In *ICLR*.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *CVPR* (pp. 2001–2010).

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *IJCV*, *115*, 211–252.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. *30*, In *NeurIPS*.

Song, Z., Zhao, Y., Shi, Y., Peng, P., Yuan, L., & Tian, Y. (2023). Learning with fantasy: Semantic-aware virtual contrastive constraint for few-shot class-incremental learning. In *CVPR* (pp. 24183–24192).

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *CVPR* (pp. 1199–1208).

Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *CVPR* (pp. 10781–10790).

Tao, X., Hong, X., Chang, X., Dong, S., Wei, X., & Gong, Y. (2020). Few-shot class-incremental learning. In *CVPR* (pp. 12183–12192).

Thongtan, T., & Phienthrakul, T. (2019). Sentiment classification using document embeddings trained with cosine similarity. In *ACL: student research workshop* (pp. 407–414).

Triantafillou, E., Zemel, R., & Urtasun, R. (2017). Few-shot learning through an information retrieval lens. *30*, In *NeurIPS*.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *JMLR*, *9*(11).

Vinyals, O., Blundell, C., Lillicrap, T., kavukcuoglu, k., & Wierstra, D. (2016). Matching networks for one shot learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *29*, *NeurIPS*.

Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). *The caltech-ucsd birds-200–2011 dataset*. California Institute of Technology.

Wang, D., Cheng, Y., Yu, M., Guo, X., & Zhang, T. (2019). A hybrid approach with optimization-based and metric-based meta-learner for few-shot learning. *Neurocomputing*, *349*, 202–211.

Wang, Y., Yao, Q., Kwok, J. T., & Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *CSUR*, *53*(3), 1–34.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., et al. (2022). Learning to prompt for continual learning. In *CVPR* (pp. 139–149).

Wang, D., Zhang, M., Xu, Y., Lu, W., Yang, J., & Zhang, T. (2021). Metric-based meta-learning model for few-shot fault diagnosis under multiple limited data conditions. *MSSP*, *155*, Article 107510.

Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *ECCV* (pp. 499–515). Springer.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., et al. (2019). Large scale incremental learning. In *CVPR* (pp. 374–382).

Yan, S., Xie, J., & He, X. (2021). Der: Dynamically expandable representation for class incremental learning. In *CVPR* (pp. 3014–3023).

Yang, S., Guo, S., Zhao, J., & Shen, F. (2024). Investigating the effectiveness of data augmentation from similarity and diversity: An empirical study. *Pattern Recognition*, *148*, Article 110204.

Yang, L., Luo, P., Change Loy, C., & Tang, X. (2015). A large-scale car dataset for fine-grained categorization and verification. In *CVPR* (pp. 3973–3981).

Yang, S., Shen, F., & Zhao, J. (2024). EntAugment: Entropy-driven adaptive data augmentation framework for image classification. In *ECCV* (pp. 197–214). Springer.

Yang, Y., Yuan, H., Li, X., Lin, Z., Torr, P., & Tao, D. (2023). Neural collapse inspired feature-classifier alignment for few-shot class incremental learning. arXiv preprint arXiv:2302.03004.

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. In *ICML* (pp. 3987–3995). PMLR.

Zhang, C., Cai, Y., Lin, G., & Shen, C. (2020). Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *CVPR* (pp. 12203–12213).

Zhang, C., Song, N., Lin, G., Zheng, Y., Pan, P., & Xu, Y. (2021). Few-shot incremental learning with continually evolved classifiers. In *CVPR* (pp. 12455–12464).

Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., et al. (2020). Class-incremental learning via deep model consolidation. In *WACV* (pp. 1131–1140).

Zhao, B., Xiao, X., Gan, G., Zhang, B., & Xia, S.-T. (2020). Maintaining discrimination and fairness in class incremental learning. In *CVPR* (pp. 13208–13217).

Zhou, K., Ethayarajh, K., Card, D., & Jurafsky, D. (2022). Problems with cosine as a measure of embedding similarity for high frequency words. arXiv preprint arXiv:2205.05092.

Zhou, D., Wang, F., Ye, H., Ma, L., Pu, S., & Zhan, D. (2022). Forward compatible few-shot class-incremental learning. In *CVPR* (pp. 9046–9056).