Continually-Adapted Margin and Multi-Anchor Distillation for Class-Incremental Learning

Yi-Hsin Chen* Dian-Shan Chen* Ying-Chieh Weng Wen-Hsiao Peng Wei-Chen Chiu
Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, Taiwan
{yhchen12101.cs09, dianshan14.c, wengyc.cs09, wpeng, walon}@nycu.edu.tw

***Abstract*— This paper addresses the problem of class-incremental learning. The model is trained to recognize the classes added incrementally. It thus suffers from the challenging issue of catastrophic forgetting. Stemming from the knowledge distillation idea of attempting to retain the model's knowledge on seen classes while learning the newly-added ones, we advance to further alleviate the catastrophic forgetting via our proposed multi-anchor distillation objective, which is realized by constraining the spatial relationship between the input data and the multiple class embeddings of each seen class in the feature space while training the model. Moreover, since the knowledge distillation for incremental learning generally relies on keeping a replay buffer to store the samples of seen classes, the buffer of limited size brings another issue of class imbalance: the number of samples from each seen class decreases gradually, thus being much smaller than the number of samples from each new class. We therefore propose to introduce the continually-adapted margin into the classification objective for tackling the prediction bias towards new classes caused by the class imbalance. Experiments are conducted on various datasets and settings to demonstrate the effectiveness and superior performance of our proposed techniques in comparison to several state-of-the-art baselines.**

I. Introduction

Deep learning have achieved great success in various vision tasks. These models are mostly trained in batch mode, with data given all at once. Recently, the needs for learning recognition models incrementally with training data arriving in a stream are emerging. For example, the single-head class-incremental learning (CIL) needs to learn newly arriving classes sequentially. Because the seen training data can only be partially retained in an experience replay buffer due to limited storage, how to learn new classes effectively while preserving the knowledge of old classes to avoid catastrophic forgetting calls for a specialized training strategy.

This paper tackles specifically single-head CIL. The training proceeds in successive incremental phases. In each phase, the model is presented with the training data of a number of new classes together with those of old classes in the replay buffer. Over incremental phases, some old training data in the replay memory are removed to make room for new data.

Prior works on single-head CIL can be grouped into two categories: parameter-based and distillation-based approaches. Parameter-based methods preserve learnt knowledge by imposing restrictions on the update of model parameters [1]–[3]. Specifically, those parameters that crucially affect the performance of old classes tend to be fixed during the learning of new classes. However, identifying those network parameters can be computationally expensive, especially with modern neural networks that are often deep and large. By contrast, distillation-based methods [4]–[10] simply require the model to produce similar predictions on old classes, an idea similar to knowledge distillation. Distillation-based methods usually involve an experience replay buffer [4] that stores partial training data of old classes. These training data are utilized together with those of new classes in updating the model to alleviate catastrophic forgetting.

While the experience replay buffer enjoys the merit of simplicity and effectiveness, the limited buffer size results in an imbalance distribution over the training data of old and new classes. The number of old training data in the replay buffer is usually much smaller than that of new classes. This yields a biased model that tends to output new classes as the prediction results. Some investigations [5], [6], [8], [11] show that the biased network weights in the fully-connected layers of the classifier are potential root causes.

A skewed decision boundary, which suggests the increased logits associated with the new classes irrespective of the input, arises as a result [12]. This observation becomes even more obvious along the incremental learning phase ; that is, the imbalance between the logits of the old and new classes increases and the prediction is biased more heavily towards new classes in a latter phase. The rationale behind these is the decreasing number of training data of old classes.

In this paper, we propose a two-pronged approach to address data imbalance and catastrophic forgetting in incremental learning. Our main contributions are: (1) We introduce a continually-adapted margin that tackles data imbalance by adapting classification logits according to the distribution of the old and new classes. (2) We apply a multi-anchor classifier to knowledge distillation in order to alleviate forgetting by preserving flexibly the relationship between the features of an input and those class anchors. (3) Extensive experiments show that our schemes achieve the state-of-art performance in various class incremental learning scenarios, striking a good balance between the learning of new classes and the knowledge preservation of old classes.

II. Related Works

Catastrophic forgetting. Knowledge distillation is a common approach to addressing catastrophic forgetting. In the context of incremental learning, it is required that the model learnt in a new incremental training phase (termed the new model) should yield similar predictions on data of old classes to the model obtained in the previous learning phase (termed

the old model). [7] is the first to introduce a modified cross-entropy loss for preserving the knowledge of the old model. [4] extends the idea and introduces a replay buffer for distillation. [9] adapts the exemplars in the replay buffer for more informative experience replay. [6] uses the distillation in feature space by requiring the feature vectors of data to have similar orientations in both old and new models. [13] applies the distillation idea at different feature levels, called Pooled Outputs Distillation Loss. By pooling features along different dimensions for distillation, their model is able to trade off better between the ability to learn new classes and forgetting. [10] utilizes the colliding effect between old and new data to reduce the size of the replay buffer. [14] splits the network into two parts to separate the learning of new classes from the distillation of old classes, followed by integrating these two parts to fine tune the model for the combined task.

Data imbalance. With replay buffer, the amount of training data for old classes is still much smaller than that for new classes. This induces a prediction bias towards new classes. Here we review techniques used to alleviate data imbalance.

– Class imbalance in CIL: [6] replaces standard softmax layer with a cosine normalization layer to balance the activations of old and new classes. Moreover, an inter-class separation loss is adopted to encourage larger separation between the new and old classes to prevent prediction bias. [8] trains a bias correction layer that adjusts the output logits to overcome prediction bias. [11] rescales the class embeddings according to their L_2 norm to avoid prediction bias toward new classes.

–Long-tailed recognition: A large body of literature has sought to address the class imbalance issue (also known as the long-tailed issue) in the general recognition tasks. The widely used techniques include (1) data resampling or loss reweighting, (2) decoupled classifier training, and (3) margin loss. Specifically, data resampling techniques over-sample the tail (scarce) classes [15] or down-sample the head (frequent) classes [15]. Reweighting methods [16], [17] balance the contributions of head and tail classes to the classification losses according to the class distribution, which reaches a similar effect to the resampling methods. Decoupled training [18] observes that the classifier is the main cause that results in the bias in imbalance training, and targets the bias correction on the classifier. Margin loss aims to increase the class separation and reduce intra-class variation. For example, [19] introduces a class-dependent margin loss, showing more balanced recognition accuracy between head and tail classes.

III. Proposed method

A. Preliminaries for class-incremental learning

We review the general setting of class-incremental learning here. Assume that there are in total N batches of training samples $\{B^1, B^2, \cdots, B^N\}$, which sequentially arrive in the corresponding N incremental learning phases, and the samples in the batch B^n belong to the new classes C^n_{new}, which are distinct from the old classes C^n_{old} that have been seen in the previous phases (i.e. $C^n_{old} = C^1_{new} \cup C^2_{new} \cup \cdots \cup C^{n-1}_{new}$). In the n-th incremental phase, the classification model learns to not only recognize C^n_{new} but also maintain its knowledge with respect to C^n_{old}, where the latter relies on a replay buffer to store the samples of C^n_{old}. Particularly, the size of the replay buffer is often limited such that not all the training data of C^n_{old} are stored. Assume the buffer has its maximum size of S samples. Typically each class in C^n_{old} has an equal number of $\bar{s}^n = \left\lfloor \frac{S}{|C^n_{old}|} \right\rfloor$ samples (named as *exemplars*) being kept in the buffer (where $|C^n_{old}|$ indicates the number of classes in C^n_{old}). Therefore, when the incremental phase proceeds with the growing $|C^n_{old}|$, the number of exemplars for each seen class decreases gradually. We denote the exemplars in the replay buffer in the n-th incremental phase as E^n.

One of the seminal works of class-incremental learning is [4], which also become the basis for our proposed method as well as various state-of-the-art methods, e.g. [5], [8], [11]. Without loss of generality, two basic loss functions (which could be modified into different variants according to different methods) are typically adopted in most class-incremental learning methods: the classification loss $\mathcal{L}_{cls}$ and the knowledge distillation loss $\mathcal{L}_{kd}$. They are detailed as follows.

Classification loss $\mathcal{L}_{cls}$ is based on the cross-entropy objective for encouraging the model to well classify both new classes C^n_{new} and old classes C^n_{old} at the same time:

$$\mathcal{L}_{cls}(x, y) = \sum_{c \in C^n_{old} \cup C^n_{new}} -\delta_{y=c} \log p_c(x), \tag{1}$$

where x is the training data $\in B^n \cup E^n$; y is the ground-truth class label of x; $p_c(x)$ denotes the posterior of class c for input x, predicted by the model; and $\delta_{y=c}$ is an indicator function.

Knowledge distillation loss $\mathcal{L}_{kd}$ extends the idea of knowledge distillation to the scenario of incremental learning. Given an input data $x \in B^n \cup E^n$, we denote its logit (i.e. the value before applying softmax $\sigma(\cdot)$ to derive the posterior) of the class $c \in C^n_{old}$ predicted by the classification model learnt in the previous incremental phase as $\hat{l}_c(x)$, and the one predicted by the model in the current phase as $l_c(x)$. The knowledge distillation loss aims to restrict the current model to produce similar predictions to the previous one, which helps preserving the model's knowledge on the seen classes:

$$\mathcal{L}_{kd}(x) = \sum_{c \in C^n_{old}} -\hat{q}_c(x) \log q_c(x), \tag{2}$$

where $\hat{q}_c(x) = \sigma(\hat{l}_c(x)/T)$ and $q_c(x) = \sigma(l_c(x)/T)$.

T here denotes the temperature scalar, thus $\mathcal{L}_{kd}$ is built upon the cross-entropy objective between softened posteriors.

B. Continually-adapted margin (CAM)

As aforementioned, since only a portion of training samples of old classes (i.e. exemplars) are kept in the replay buffer, the number of exemplars per old class $\in C^n_{old}$ is typically much less than the number of training samples

per new class $\in C^n_{new}$, which leads to the problem of **class imbalance**. In particular, such class imbalance between C^n_{old} and C^n_{new} becomes more severe while incremental learning proceeds, as the number of exemplars per old class is getting smaller. Basically, during test time, the classification model trained on the dataset with the class imbalance issue is prone to give the new classes (having more training samples) higher logits, while the old classes (having fewer training samples) tend to get lower logits. Such bias toward new classes can also be connected to the phenomenon of having a skewed decision boundary [12]. In order to tackle this bias as the main cause of catastrophic forgetting for the class-incremental learning method with replay buffer, we introduce the **continually-adapted margin (CAM)**. By adding a margin Δ^n_c to each class c, we adjust the corresponding logit predicted by the classification model in order to rectify the skewed decision boundary, where such margin Δ^n_c is adaptive according to the number of accessible training samples (denoted as ε^n_c) for the class c in the n-th incremental phase. Given a training sample x, its posterior $p_c(x)$ for class c after applying our continually-adapted margin (denoted as $\tilde{p}_c(x)$) is defined as:

$$\tilde{p}_c(x) = \frac{e^{l'_c(x)-\Delta^n_c}}{\sum_{j\in C^n} e^{l'_j(x)-\Delta^n_j}} \quad , \quad \Delta^n_c = \gamma \cdot (\frac{\bar{s}^n}{\varepsilon^n_c})^{\frac{1}{4}} \tag{3}$$

where $C^n = C^n_{old} \cup C^n_{new}$, and γ is a hyper-parameter used to control the maximal margin among all the classes C^n. Note that the margin is only added in the training stage and the logit $l'_c(x)$ used here is computed on the multiple class embeddings, which we will detail in the next subsection. Such rectified posterior $\tilde{p}_c(x)$ is adopted to replace $p_c(x)$ of $\mathcal{L}_{cls}$ defined in Eq. (1), which results in the classification loss $\mathcal{L}_{\text{CAM}}$ used to train our model:

$$\mathcal{L}_{\text{CAM}}(x, y) = \sum_{c\in C^n} -\delta_{y=c} \log \tilde{p}_c(x), \tag{4}$$

We remark that the design of our margin Δ^n_c is inspired by the margin loss proposed in [19], which was initially aimed at tackling the long-tailed recognition. However, a direct application of [19] to our class-incremental learning problem does not yield good performance. We thus adapt its design in two ways. First, different from [19], where the margin Δ^n_c is given to the logit of the positive class (i.e. the ground-truth class) only, our scheme attaches margin to all the classes, positive or negative. This more balanced design is found beneficial to not only class-incremental learning but also long-tailed recognition. Second, with a different application from [19], which addresses long-tailed recognition under the non-incremental learning setup, our margin Δ^n_c is specifically chosen to suit class-incremental learning in a way that the margins for old classes remain fixed across incremental learning phases. This is meant to preserve the learnt knowledge, in order to mitigate catastrophic forgetting. In Eq. (3), the margins Δ^n_c for old classes $c \in C^n_{old}$ evaluate to γ, which is a constant throughout the whole training process because the accessible training samples ε^n_c for old classes are always the $\bar{s}^n$ exemplars (cf. Section III-A) in the replay buffer. As such, $\varepsilon^n_c = \bar{s}^n$ and $\Delta^n_c = \gamma$. Moreover, to address the issue that the class imbalance gets more skewed along the incremental phases (as $\bar{s}^n$ gradually decreases together with the increasing $|C^n_{old}|$), the margin has the desirable property that it becomes increasingly smaller for new classes as the incremental phase increases. For new classes $c \in C^n_{new}$, because their available training samples ε^n_c are typically more than the ones of old classes, Δ^n_c is then smaller than γ. To sum up, the old classes have larger margins than the new classes; the gap between their margins grows as the incremental phase increases, the process of which ensures that the classification model yields higher logit values for old classes to minimize $\mathcal{L}_{\text{CAM}}$.

Please note that some existing approaches of class-incremental learning (e.g. [8], [11]) also propose to correct the classification bias towards new classes caused by the class imbalance, via various manners. In experiments, we compare our CAM with these prior works to validate its effectiveness.

C. Multi-anchor knowledge distillation (MAKD)

Without loss of generality, given an input data x, its logit $l_c(x)$ with respect to the class c produced by the typical classification model can be written as $\phi(x)^\top w_c$, i.e. the inner product between the feature of x extracted by the feature extractor $\phi(\cdot)$ and the class embedding w_c of the class c contained in the fully connected layer. In such a scenario, each class c is assumed to have only a single embedding w_c, which might not be capable of well handling the class with multi-modal feature distributions. We hypothesize that such multi-modal feature distributions, which are attributed to the large intra-class variability, are common in real-world datasets. To this end, we introduce K embeddings $\{w^k_c\}^K_{k=1}$ for each class c, named as **multiple anchors** in our work. Then, the logit $l'_c(x)$ based on these anchors is computed as follows:

$$\begin{aligned} l'_c(x) &= \phi(x)^\top \mathbf{w}_c \\ \text{where } a^k_c = \frac{e^{\phi(x)^\top w^k_c/\tau}}{\sum^K_{j=1} e^{\phi(x)^\top w^j_c/\tau}} &\text{ and } \mathbf{w}_c \quad = \sum^K_{k=1} a^k_c w^k_c \end{aligned} \tag{5}$$

and τ is a temperature scalar, called the **intra-class temperature**, used to soften the contribution of each anchor for deriving the aggregated anchor of the class c (i.e. $\mathbf{w}_c$). Such aggregation is based on the similarities between the input feature $\phi(x)$ and the multiple anchors $\{w^k_c\}^K_{k=1}$, thus being data-dependent. Also, the aggregated anchor $\mathbf{w}_c$ is actually located within the simplex built on $\{w^k_c\}^K_{k=1}$. Note that the logit $l'_c(x)$ with multiple anchors is used for computing $\tilde{p}_c(x)$ in $\mathcal{L}_{\text{CAM}}$.

Moreover, based on these multiple anchors, we are able to extend the basic knowledge distillation loss $\mathcal{L}_{kd}$ defined in Eq (2) to be the $\mathcal{L}_{\text{MAKD}}$ objective of **multi-anchor knowledge distillation (MAKD)** used to train our model:

$$\mathcal{L}_{\text{MAKD}}(x) = \sum_{c\in C_{old}} -\sigma(\hat{l}'_c(x)/T) \log \sigma(l'_c(x)/T) \tag{6}$$

where $l'_c(x)$ and $\hat{l}'_c(x)$ denote the logits of class c predicted by the classification model in the current phase and the

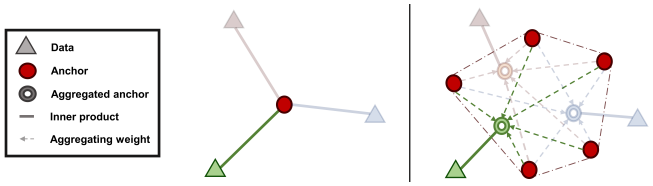


Fig. 1. Illustration of spatial relationship in the feature space (related to class posteriors) used by [left] the basic knowledge distillation loss $\mathcal{L}_{kd}$ and [right] our multi-anchor knowledge distillation loss $\mathcal{L}_{\text{MAKD}}$, where the former has a single embedding/anchor per class while the latter has multiple anchors. For $\mathcal{L}_{kd}$, a fixed anchor is used to compute the similarity with respect to all samples, while for $\mathcal{L}_{\text{MAKD}}$ an aggregated anchor is adopted. Particularly, the combination over multiple anchors to obtain the aggregated anchor is data-dependent (cf. Eq. (5)), where the aggregated anchor is within the simplex (outlined by dash-dotted lines) built upon multiple anchors.

one learnt in the previous incremental phase, respectively, in which the computation of $\hat{l}'_c(x)$ follows the same procedure as Eq. (5) but uses both the feature extractor $\hat{\phi}(\cdot)$ and multiple anchors $\{\hat{w}_c^k\}_{k=1}^K$ obtained from the previous incremental phase. The temperature T here is the **inter-class temperature**, which is to be distinguished from the intra-class temperature.

As shown in Figure 1, our multi-anchor knowledge distillation loss $\mathcal{L}_{\text{MAKD}}$ considers the spatial relationship between samples and the "data-dependent" aggregated anchor $\mathbf{w}_c$ of each class ([right] in Figure 1), where it not only adopts the higher-order spatial constraints (i.e. versus multiple anchors) but also provides more flexibility due to the data-dependent property of $\mathbf{w}_c$. Later in the evaluation section, our $\mathcal{L}_{\text{MAKD}}$ loss is experimentally shown to not only lower the forgetting, but also improve the accuracy. In passing, the main difference between our MAKD and Local Similarity Classifier in [13], which applies Eq. (5) to classification only, is that our MAKD uses Eq. (5) for both classification and knowledge distillation.

Our overall objective function is summarized as:

$$\mathcal{L}(x, y) = (1 - \lambda)\mathcal{L}_{\text{CAM}}(x, y) + \lambda\mathcal{L}_{\text{MAKD}}(x) \qquad (7)$$

where $\lambda = |C_{old}^n|/(|C_{old}^n| + |C_{new}^n|)$ in the n-th incremental phase to control the trade-off between $\mathcal{L}_{\text{CAM}}$ and $\mathcal{L}_{\text{MAKD}}$. Along with incremental phases, $|C_{old}^n|$ increases thus the overall objective tends to strengthen $\mathcal{L}_{\text{MAKD}}$ for further enhancing the preservation of model's knowledge on seen classes.

IV. Experimental results

Datasets. We adopt CIFAR-100 and ImageNet datasets for evaluating various methods of class-incremental learning. We follow the experimental setting as [4] to sample 100 classes from ImageNet to form a subset, denoted as ImageNet-sub.

Training schemes. There are two common training schemes: *training-from-scratch* (TFS) and *training-from-half* (TFH). For TFS, the model is trained from scratch without any pre-training. In practice, the classes in the dataset are equally divided into N packages, which are used sequentially in N incremental phases for training the model. For TFH, half of the classes in the dataset are firstly used to pre-train the feature extractor of the classification model (i.e. the model starts class-incremental learning with a strong feature extractor), and then the remaining half of the classes are equally divided into N packages to be learned sequentially in N incremental phases. For the replay buffer, it can store up to S exemplars (S = 2000 for all our experiments unless otherwise stated). For each of the old classes, the exemplars to be kept in the replay buffer are selected via the herding strategy as [4].

Evaluation metrics. We follow the evaluation metrics proposed by [4], which include the *incremental accuracy*, the *average incremental accuracy,* and the *average forgetting measure* [3]. For the first two metrics, the higher the better, whereas for the forgetting measure, the lower the better.

Implementation details. We follow iCaRL [4] to employ a 32-layer (respectively 18-layer) ResNet [20] as the feature extractor for the experiments on CIFAR-100 (respectively ImageNet-sub) dataset. We train the model using the SGD optimizer with a momentum of 0.9 and set the initial learning rate to 0.1 for all experiments. For the CIFAR-100 dataset, each incremental phase has 250 epochs, with learning rate decay being 0.1 after 100, 150, and 200 epochs. The weight decay is set to 0.0005 and the batch size is 128. The intra-class temperature τ in $\mathcal{L}_{\text{MAKD}}$ is set to 10. While for the ImageNet-sub dataset, each incremental phase has 100 epochs, with learning rate decay being 0.1 after 30, 60, 80, and 90 epochs. The weight decay is set to 0.0001 and the batch size is 256. The intra-class temperature τ in $\mathcal{L}_{\text{MAKD}}$ is set to 1. We adopt random cropping and horizontal flip for all datasets as iCaRL [4]. For all experiments, the number of anchors K per class is set to 10, the maximal margin γ used in CAM is set to 7, and the inter-class temperature T in $\mathcal{L}_{\text{MAKD}}$ is set to 2. We will release our code and models after paper acceptance.

A. Quantitative comparison

We compare our proposed method with several state-of-the-art or representative baselines of class-incremental learning, including three methods (i.e. iCaRL [4], BiC [8], WA [11]) originally proposed in the TFS scheme and another four methods (i.e. LUCIR [6], Mnemonics [9], PODNet [13], DDE [10][1]) designed for the TFH scheme. Table I summarizes the results on two datasets, two training schemes, and various settings of the number of incremental phases N, by three evaluation metrics. Please note that we use 5 random orderings of classes to construct incremental phases, and all performance numbers are computed over these 5 orderings.

We observe that, among all the different settings, our proposed method achieves the best or the second best performance. Moreover, we see that these baseline methods have strong preference on their specific training schemes. For instance, with the TFH scheme, the most competitive baselines to ours are {LUCIR, Mnemonics, PODNet, DDE}, but their performance turn out to be much worse under the

[1]Official released code of DDE contains only the version with LUCIR, thus results of this version is provided.

TABLE I

COMPARISON AMONG VARIOUS METHODS BASED ON BOTH CIFAR-100 AND IMAGENET-SUB DATASETS UNDER TWO TRAINING SCHEMES WITH DIFFERENT TOTAL NUMBER OF INCREMENTAL PHASES: $N = \{5, 10, 20\}$ FOR TRAINING-FROM-SCRATCH (TFS) WHILE $N = \{5, 10\}$ FOR TRAINING-FROM-HALF (TFH). WE ADOPT THE INCREMENTAL ACCURACY AT THE END OF THE ENTIRE INCREMENTAL LEARNING (ABBREVIATED AS *last* (↑)), THE AVERAGE INCREMENTAL ACCURACY (ABBREVIATED AS *avg.* (↑)), AND THE AVERAGE FORGETTING MEASURE (ABBREVIATED AS *forg.* (↓)) AS THE EVALUATION METRICS. **RED** AND GREEN INDICATE THE BEST AND THE SECOND BEST PERFORMANCE.

Method	TFS									TFH					
	$N = 5$			10			20			5			10		
	last	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*
CIFAR-100															
iCaRL	50.1	60.9	24.1	43.1	55.1	20.5	37.7	49.2	26.8	52.4	62.0	20.3	49.4	59.1	25.0
LUCIR	48.1	63.2	32.1	39.9	53.5	39.4	36.5	54.1	45.2	54.5	64.0	28.6	48.8	59.3	32.2
Mnemonics	45.1	58.7	31.6	40.9	53.4	33.0	39.1	50.2	31.6	54.2	63.3	18.8	**55.0**	**63.5**	16.9
PODNet	48.1	62.1	31.2	38.5	53.1	39.4	31.5	44.3	46.3	54.2	64.0	22.6	51.9	62.0	25.4
DDE	49.3	62.0	**13.5**	39.9	54.3	**14.2**	33.5	45.5	**12.0**	55.0	64.3	**8.1**	50.7	61.3	**11.8**
BiC	53.4	65.2	24.5	44.0	59.0	28.6	37.8	55.0	34.9	52.4	62.4	26.2	46.6	56.2	30.4
WA	58.2	68.5	18.0	51.9	65.7	23.9	41.6	60.0	37.6	54.0	62.6	24.4	43.1	54.6	38.5
Ours	**58.5**	**69.2**	19.3	**54.0**	**67.1**	21.4	**48.5**	**63.9**	21.4	**56.8**	**65.1**	17.5	52.7	62.3	16.9
ImageNet-sub															
iCaRL	60.7	71.5	20.0	55.5	67.6	20.2	47.9	62.1	28.5	60.7	68.9	20.2	52.8	63.2	29.1
LUCIR	49.4	66.2	39.1	34.4	50.9	48.5	32.4	52.2	57.1	60.1	69.2	29.2	51.4	63.1	34.6
Mnemonics	48.2	61.1	20.0	39.4	49.2	31.5	38.6	43.0	30.0	61.0	69.4	13.7	**61.4**	**69.1**	12.8
PODNet	58.7	71.4	23.9	47.2	61.9	36.3	35.6	50.4	48.6	64.2	**73.0**	20.8	59.5	68.7	27.1
DDE	54.5	67.0	15.0	42.4	56.5	21.0	34.8	47.3	25.5	62.4	70.5	**8.8**	59.3	68.0	11.4
BiC	61.0	71.4	25.5	56.1	69.6	31.0	46.9	63.8	42.3	59.7	68.7	27.3	51.2	61.6	38.2
WA	57.3	68.2	30.5	49.2	62.7	37.1	42.7	57.7	43.1	53.8	63.6	38.0	47.0	57.4	32.1
Ours	**65.3**	**73.2**	**13.6**	**59.3**	**69.7**	**16.8**	**50.0**	**63.9**	**18.4**	**67.5**	**73.0**	10.1	60.8	68.7	**11.3**

TABLE II

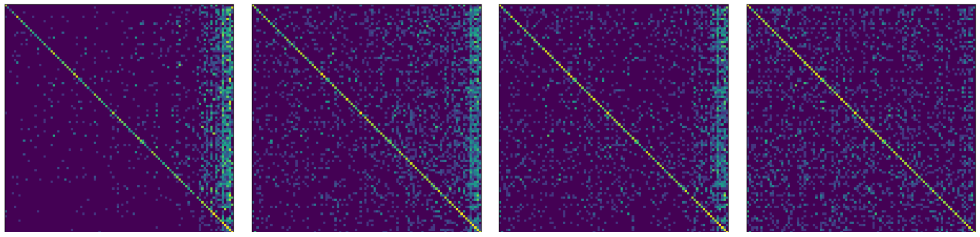
COMPARISON AMONG VARIOUS LONG-TAILED METHODS ON CIFAR-100 UNDER TWO TRAINING SCHEMES WITH DIFFERENT TOTAL NUMBER OF INCREMENTAL PHASES: $N = \{5, 10, 20\}$ FOR TFS WHILE $N = \{5, 10\}$ FOR TFH. WE ADOPT THE LAST INCREMENTAL ACCURACY (ABBREVIATED AS *last* (↑)), THE AVERAGE INCREMENTAL ACCURACY (ABBREVIATED AS *avg.* (↑)), AND THE AVERAGE FORGETTING MEASURE (ABBREVIATED AS *forg.* (↓)) AS THE EVALUATION METRICS. **RED** AND GREEN INDICATE THE BEST AND THE SECOND BEST PERFORMANCE.

Method	TFS									TFH					
	$N = 5$			10			20			5			10		
	last	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*	*last*	*avg.*	*forg.*
`Base`	45.1	62.5	44.2	36.0	56.5	51.8	29.9	51.4	57.8	40.4	53.8	50.2	34.9	48.5	54.9
`Base`+RS	52.8	66.0	29.0	45.5	62.1	29.7	40.4	59.3	**24.5**	49.6	60.7	25.8	43.8	56.0	24.1
BiC	53.4	65.2	24.5	44.0	59.0	28.6	37.8	55.0	34.9	52.4	62.4	26.2	46.6	56.2	30.4
WA	**58.2**	68.5	**18.0**	51.9	65.7	**23.9**	41.6	60.0	37.6	54.0	62.6	24.4	43.1	54.6	38.5
`Base`+CAM	57.8	**69.0**	21.3	**52.0**	**65.9**	24.7	**44.5**	**60.6**	25.3	**56.4**	**65.6**	**19.8**	**50.4**	**61.5**	**20.4**
Ours	58.5	69.2	19.3	54.0	67.1	21.4	48.5	63.9	21.4	56.8	65.1	17.5	52.7	62.3	16.9

TFS scheme. This observation indicates that these methods highly rely on having a strong feature extractor to start with in order to achieve good performance for the task of class-incremental learning. On the other hand, {BiC, WA} perform closely to our scheme in the TFS scheme, but are much worse than ours in the TFH scheme. In comparison, our proposed method shows consistently superior performance in both training schemes across CIFAR-100 and ImageNet-sub datasets, confirming its superiority and generalizability.

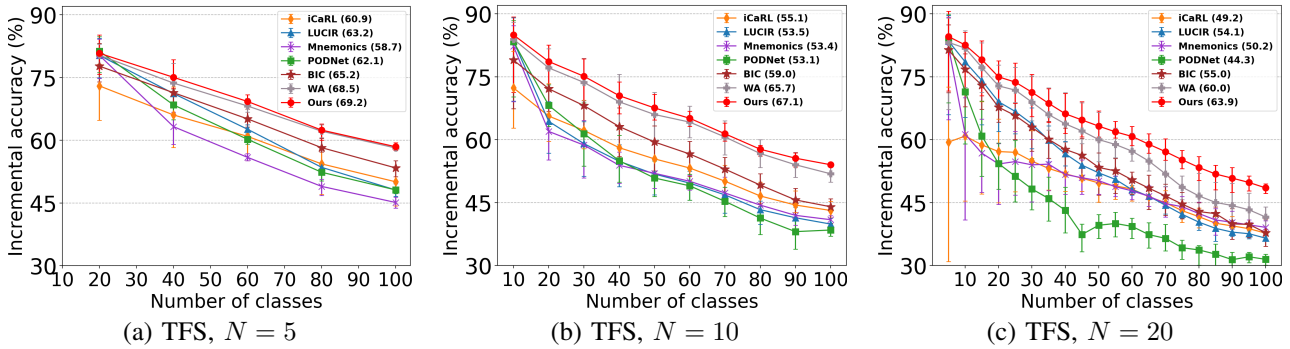
Confusion matrices comparison. Figure 2 visualizes the confusion matrices of classification obtained from `Base`, BiC, WA, and our full model. `Base` is simply trained by using $\mathcal{L}_{cls}$ and $\mathcal{L}_{kd}$; it does not particularly address the class imbalance problem. In contrast, BiC and WA tackle the class-imbalance issue by correcting the bias related to the fully-connected classifier at the end of each incremental phase. As seen, our proposed method shows much less bias towards the new classes, suggesting its superior capability of addressing the class imbalance problem.

Incremental accuracy. Figure 3 and Figure 4 plot the incremental accuracy along the incremental learning phases for CIFAR-100 and ImageNet-sub datasets, respectively. The results presented include both training-from-scratch (TFS) and training-from-half (TFH) schemes and a variety of incremental learning phases N. The average incremental accuracy of each method is shown in parentheses. Note that each point in the plots is averaged over five orderings of classes and an error bar is used to indicate the standard

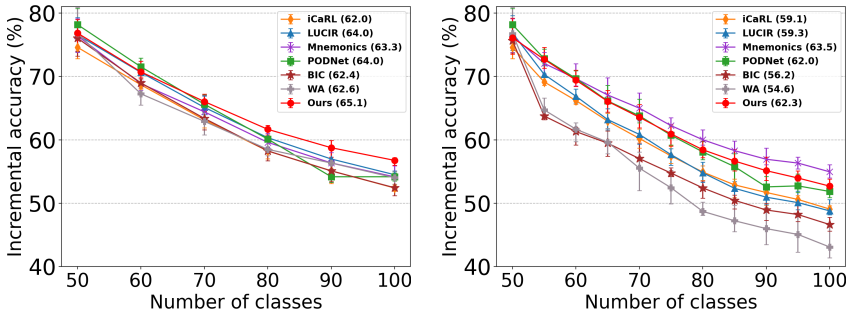


(a) Base (b) BiC (c) WA (d) Ours

Fig. 2. Comparison between various methods in terms of confusion matrix (where all entries v in the matrix are transformed by $\log(1+v)$ for better visualization). Experiments are conducted on the CIFAR-100 dataset under the scheme of TFS with the number of incremental phases $N = 20$. These matrices show that the baseline methods (e.g. Base, BiC [8], and WA [11]) still suffer from the class imbalance problem as their resultant classification models have significant bias (related to large confusion) towards the new classes being added in the last few incremental phases.



(a) TFS, $N = 5$ (b) TFS, $N = 10$ (c) TFS, $N = 20$



(d) TFH, $N = 5$ (e) TFH, $N = 10$

Fig. 3. Incremental accuracy on CIFAR-100 dataset with TFS (top row) and TFH (bottom row). The average incremental accuracy of each method is shown in parentheses. In most of the settings, our method achieves the highest incremental accuracy at the completion of the entire incremental training (see the rightmost point of each curve).

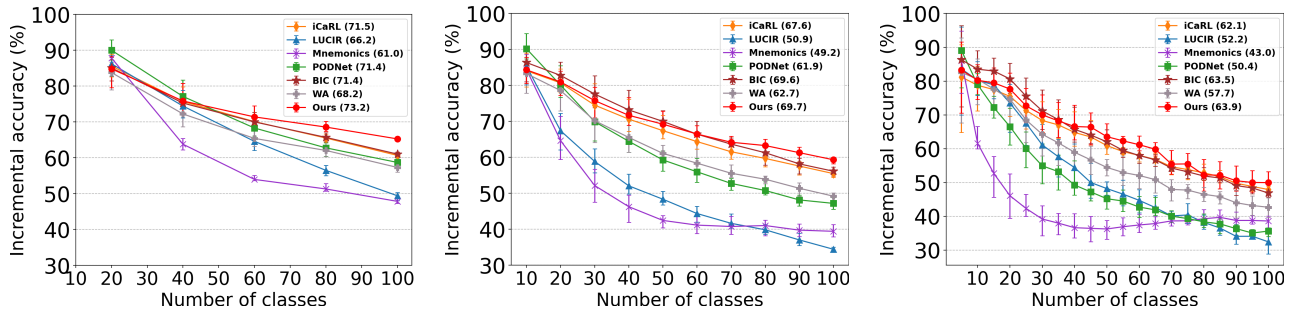
deviation. In most of the settings, our method achieves the best incremental accuracy at the end of incremental training (i.e. the highest accuracy at the rightmost point in each plot). On ImageNet-sub dataset, the baseline methods show better incremental accuracy than ours in the first few incremental phases; however, their performances drop quickly with the increasing number of learnt classes. The result suggests that they suffer from catastrophic forgetting. In comparison, the performance of our method decreases at a relatively slower rate, showing that our method is more robust to forgetting. This observation is in line with the results in Table I, where our method shows the lowest average forgetting measure in most of the test cases.

B. Ablation study

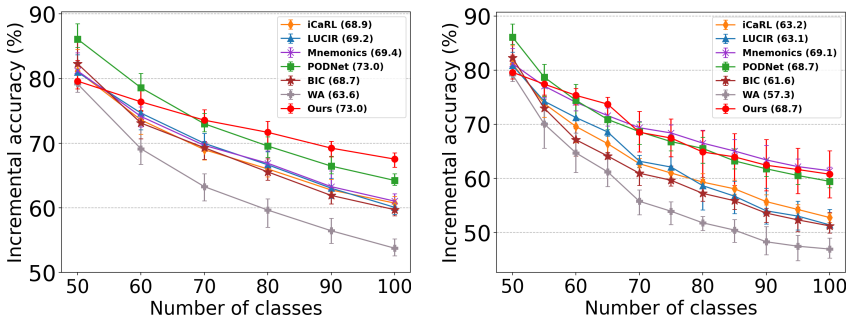
The distribution of training data versus the CAM. We visualize how (1) the number of training data per class and (2) the value of our continually-adapted margin Δ (in our proposed CAM technique) vary with the incremental learning phase. The results are reported for the CIFAR-100 dataset and under the TFS scheme.

Figure 5 shows that for those old classes, the number of per-class training data (the red curve) decreases with the incremental learning phase. This is because of the limited size ($S = 2000$) of the replay buffer. With the increasing number of seen classes, the number of exemplars that can be stored for each seen class decreases. On the other hand, the number of training data of each new class is maintained at a fixed level (the blue curve), i.e. 500 training samples for CIFAR-100 dateset. The increasing gap between the red and blue lines along the incremental phase indicates the increasing imbalance between the training data of the old and new classes.

Figure 6 presents the resulting change of margin in our CAM design. As described in the main paper, when the class



(a) TFS, $N = 5$ (b) TFS, $N = 10$ (c) TFS, $N = 20$



(d) TFH, $N = 5$ (e) TFH, $N = 10$

Fig. 4. Incremental accuracy on ImageNet-sub dataset with TFS (top row) and TFH (bottom row). The average incremental accuracy of each method is shown in parentheses. Our method obtains a less steep curve as compared with the other baseline methods, which indicates less forgetting across the incremental learning phases.

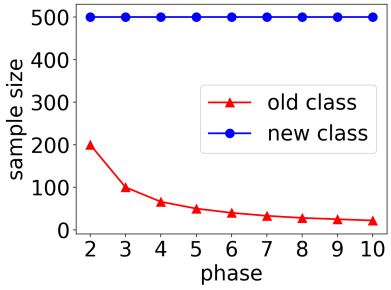


Fig. 5. Demonstrates that the number of exemplars per old class decreases with the increasing learning phase due to the fixed-size of replay buffer, thus increases the data imbalance between the old and new classes.

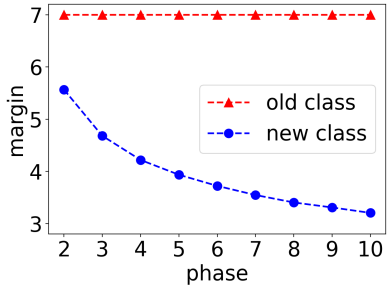


Fig. 6. Shows that our continually-adapted margin (CAM) is able to adapt the margin Δ of new classes to address the increased data imbalance along the learning phases.

c belongs to C^n_{old}, the margin Δ^n_c is equal to γ (and $\gamma = 7$ here). According to Figure 5, the new classes usually have more training data than the old classes. Therefore, the term $(\bar{s}^n/\varepsilon^n_c)^{\frac{1}{4}}$ in our definition of the CAM becomes less than 1 (as $\varepsilon^n_c > \bar{s}^n$), the margin of each new class thus become less than γ (the blue curve in Figure 6). The growing gap between the margins of the old and new classes indicates that our continually-adapted margin can adapt the margin of new classes to address the increased data imbalance along the learning phases. In addition, the margin value in our paper could be further explored. There are many factors in incremental learning that affect the optimal value for margins (e.g. the total number of incremental phases N, training schemes, and the maximal size of exemplars S). We believe that there could exist more suitable design of margin for the class-incremental learning to strive for a better class balance and leave this exploration for future work.

Data imbalance: First, we investigate how the three types of long-tailed recognition methods–namely, data resampling, decoupled classifier training, and margin loss–perform in terms of addressing the data imbalance issue in class-incremental learning. We experiment these methods on a base model (`Base`), which is trained with $\mathcal{L}_{cls}$ (Eq. (1)) and $\mathcal{L}_{kd}$ (Eq. (2)). In the following, `Base`+RS refers to the data resampling method, where the old class data are resampled from the accessible exemplars to match the amount of new class data; WA [11] and BiC [8] are methods with the decoupled classifier training, where the training is done in two stages with the bias correction applied to the classifier in the second stage; and `Base`+CAM represents the margin loss methods. We evaluate their performance on CIFAR-100, under the TFS and TFH schemes for a varied number of N. As seen in Table II, `Base`+CAM performs among the top-2 across all settings in terms of all three evaluation metrics. Additionally, the results of our full model shows that `Base`+CAM, when extended to our full model, can be fur-

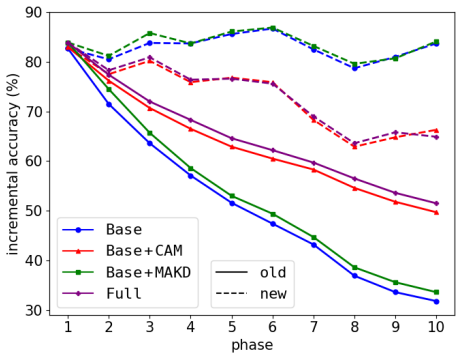


Fig. 7. Comparison on different model variants to study the contributions of CAM and MAKD to the accuracy on new and old classes. Experiments are based on CIFAR-100 dataset and the TFS scheme, with $N = 10$.

ther improved by the proposed MAKD. In comparison, WA cannot benefit from MAKD since it requires the classifier to be a single fully-connected layer.

Accuracy vs. forgetting: In Table II, a comparison between `Base`+CAM and Ours shows that MAKD improves accuracy while mitigating the forgetting. Often we have one method performing better than another in terms of accuracy while showing worse forgetting, or the vice versa. One example is iCaRL [4] vs. BiC [8] in Table I.

Contributions of CAM and MAKD: In this section, we conduct an ablation experiment to analyze the individual contributions of CAM and MAKD, via comparison among several variants of our model: (a) `Base` trained with the classification loss $\mathcal{L}_{cls}$ and knowledge distillation loss $\mathcal{L}_{kd}$; (b) `Base`+CAM extends from `Base` to add CAM for computing the posteriors used in $\mathcal{L}_{cls}$. In this variant, there is only one anchor per class (i.e. MAKD is disabled); (c) `Base`+MAKD introduces multi-anchors into both $\mathcal{L}_{cls}$ and $\mathcal{L}_{kd}$, which becomes $\mathcal{L}_{\text{MAKD}}$. In this variant, CAM is disabled; and (d) `Full` indicates our full model, where both $\mathcal{L}_{\text{CAM}}$ and $\mathcal{L}_{\text{MAKD}}$ are used. The experiment is conducted on CIFAR-100 under the TFS scheme with $N = 10$. In Figure 7, the incremental accuracy on new and old classes is presented in pairs of dashed and solid lines. The results of different variants are visualized in separate colors. From the figure, two main observations are made. (1) `Base` vs. `Base`+CAM: it is seen that the gap between the blue lines is much larger than that between the red lines, which suggests CAM is able to achieve more balanced accuracy on new and old classes. The same observation holds true when MAKD is present, by comparing the gap between the green lines (`Base`+MAKD) with that between the purple lines (Full). (2) `Base` vs. `Base`+MAKD: the comparison of the blue and green solid lines shows that MAKD is able to improve the accuracy on old classes while maintaining similar performance on new classes (the blue and green dashed lines). The same observation carries over when CAM is enabled.

V. Conclusions

In this work of incremental learning, to address the class imbalance issue which gets more serious as incremental phases proceed, we introduce CAM, which continuously adjusts the margins for all classes in order to accommodate different degrees of class imbalance over incremental phases. We also propose MAKD, which aims to preserve both the higher-order and data-dependent spatial relationships between the features of samples and the class embeddings across incremental phases, thereby maintaining the knowledge and the discrimination of the seen/learnt classes. Experiments show that these simple yet critical solutions improve both accuracy and forgetting metrics across various settings and datasets.

Acknowledgement This work is supported by NSTC 111-2628-EA49-018-MY4 & MOST 110-2221-E-A49-065-MY3.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *ECCV*, 2018.

[2] Friedemann Zenke, Ben Poole, and Surya Ganguli, "Continual learning through synaptic intelligence," *JMLR*, 2017.

[3] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *ECCV*, 2018.

[4] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert, "icarl: Incremental classifier and representation learning," in *CVPR*, 2017.

[5] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari, "End-to-end incremental learning," in *ECCV*, 2018.

[6] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin, "Learning a unified classifier incrementally via rebalancing," in *CVPR*, 2019.

[7] Zhizhong Li and Derek Hoiem, "Learning without forgetting," *TPAMI*, 2017.

[8] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu, "Large scale incremental learning," in *CVPR*, 2019.

[9] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun, "Mnemonics training: Multi-class incremental learning without forgetting," in *CVPR*, 2020.

[10] Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang, "Distilling causal effect of data in class-incremental learning," in *CVPR*, 2021.

[11] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia, "Maintaining discrimination and fairness in class incremental learning," in *CVPR*, 2020.

[12] Salman Khan, Munawar Hayat, Syed Waqas Zamir, Jianbing Shen, and Ling Shao, "Striking the right balance with uncertainty," in *CVPR*, 2019.

[13] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle, "Podnet: Pooled outputs distillation for small-tasks incremental learning," in *ECCV*, 2020.

[14] Jong-Yeong Kim and Dong-Wan Choi, "Split-and-bridge: Adaptable class incremental learning within a single neural network," *ArXiv:2107.01349*, 2021.

[15] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, 2018.

[16] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert, "Learning to model the tail," in *NeurIPS*, 2017.

[17] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie, "Class-balanced loss based on effective number of samples," in *CVPR*, 2019.

[18] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis, "Decoupling representation and classifier for long-tailed recognition," *ArXiv:1910.09217*, 2019.

[19] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," *ArXiv:1906.07413*, 2019.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.