Maintaining Discrimination and Fairness in Class Incremental Learning

Bowen Zhao[†,‡] Xi Xiao[†,‡] Guojun Gan[*] Bin Zhang[‡] Shutao Xia[†,‡]
[†]Tsinghua University [‡]Peng Cheng Laboratory [*]University of Connecticut
zbw18@mails.tsinghua.edu.cn, {xiaox,xiast}@sz.tsinghua.edu.cn
bin.zhang@pcl.ac.cn, guojun.gan@uconn.edu

Abstract

Deep neural networks (DNNs) have been applied in class incremental learning, which aims to solve common real-world problems of learning new classes continually. One drawback of standard DNNs is that they are prone to catastrophic forgetting. Knowledge distillation (KD) is a commonly used technique to alleviate this problem. In this paper, we demonstrate it can indeed help the model to output more discriminative results within old classes. However, it cannot alleviate the problem that the model tends to classify objects into new classes, causing the positive effect of KD to be hidden and limited. We observed that an important factor causing catastrophic forgetting is that the weights in the last fully connected (FC) layer are highly biased in class incremental learning. In this paper, we propose a simple and effective solution motivated by the aforementioned observations to address catastrophic forgetting. Firstly, we utilize KD to maintain the discrimination within old classes. Then, to further maintain the fairness between old classes and new classes, we propose Weight Aligning (WA) that corrects the biased weights in the FC layer after normal training process. Unlike previous work, WA does not require any extra parameters or a validation set in advance, as it utilizes the information provided by the biased weights themselves. The proposed method is evaluated on ImageNet-1000, ImageNet-100, and CIFAR-100 under various settings. Experimental results show that the proposed method can effectively alleviate catastrophic forgetting and significantly outperform state-of-the-art methods.

1. Introduction

In the past few years, Deep Neural Networks (DNNs) have shown remarkable performance in various applications, even surpassing human performance on some tasks [10, 11, 16]. The standard DNNs are typically trained on a prepared dataset, where the number of categories is fixed in advance. However, in many real-world applications, it is often required to learn new classes gradually from streaming

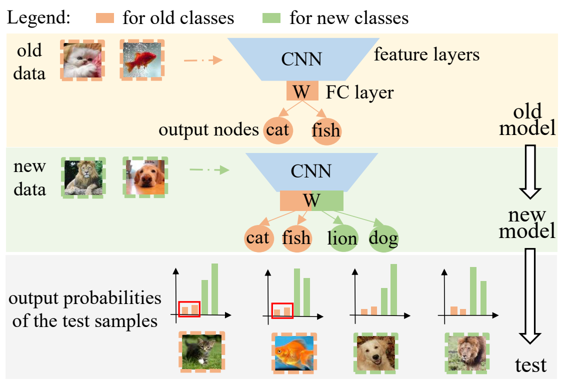


Figure 1: A vanilla method for class incremental learning.

data, which is called class incremental learning.

In order to achieve this goal, a common method is to fine tune the old model on new data by setting the number of output nodes to be that of current classes (including old and new classes) as shown in Figure 1. However, this naive method suffers from a serious problem known as catastrophic forgetting [7, 22]. As can be seen from Figure 1, the old data's output probabilities corresponding to the old classes (which are highlighted in red boxes) are relatively low. Thus, the new model trained by the vanilla method generally predicts objects as new classes [26, 33, 39].

To alleviate catastrophic forgetting, many studies have been done [28]. EWC [18], SI [35], and MAS [1] attempt to solve this problem with a parameter control strategy. Knowledge distillation (KD) [12] is another strategy, which has also been widely used in this field [5, 20, 40]. Besides, some other studies [23, 26, 29, 34] follow a rehearsal strategy by using a small amount of real or generated old data in the training process. In class incremental learning tasks, the new model is trained without access to the old data, even with the rehearsal strategy, the training set in an incremental step is seriously imbalanced between old classes and new classes. Thus, there are also some studies that deal with catastrophic forgetting from this perspective [13, 33].

In this paper, we demonstrate that knowledge distillation, the commonly used technique in this field, can in-

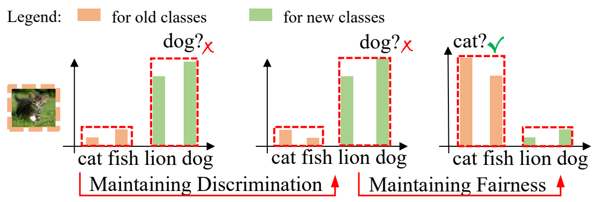


Figure 2: The effect of our solution. KD helps model to maintain discrimination within old classes. WA helps model to maintain fairness between old and new classes.

deed help the model to output more discriminative results within old classes. However, the prediction bias towards new classes cannot be alleviated. The trained model still treats old classes unfairly, causing the positive effect of KD to be hidden and limited. Then we show that the weights in the trained model's FC layer are heavily biased, which can cause the model to tend to classify samples into new classes. Based on the above, we present a simple and effective solution to mitigate catastrophic forgetting. The effect of our solution is presented in Figure 2. Firstly, we utilize KD to maintain the discrimination within old classes. This helps the model to output more discriminative results within old classes. Then, to further maintain the fairness between old and new classes, we propose Weight Aligning (WA) that corrects the biased weights in the FC layer after the normal training process. This helps the model to treat old classes and new classes fairly, and output correct predictions.

In this paper, our main contributions are the following: (i) We investigated the actual role of KD in class incremental learning by experiments; (ii) We presented a simple and effective solution to address catastrophic forgetting in class incremental learning that maintains both the discrimination via KD and the fairness via WA; (iii) Inspired by a prior observation of a non-incremental model, the proposed method WA attempts to align the norms of the weight vectors for new classes to those for old classes. WA makes full use of the information contained in the trained model and correct the biased weights in the FC layer, it does not need to reserve a validation set in advance or require any additional parameters to be tuned, but can handle class incremental learning tasks well; (iv) Extensive experiments were conducted, the results show that our method achieves better performance than previous methods.

2. Related Work

Recently, many methods have been proposed to alleviate the well-known problem of catastrophic forgetting [7, 22] suffered by ordinary DNNs . In this section, we briefly discuss these methods.

Parameter Control. The approaches of this strategy such as EWC [18], SI [35], and MAS [1] manage to constrain the important weights of old model when facing new data. These methods expect small changes in the important parameters. They differ in how to estimate the important parameters. EWC estimates the weight importance through the Fisher information matrix; SI uses the path integral over the optimization trajectory; MAS utilizes the gradients of the network output [38]. However, the importance of parameters is difficult to measure accurately in a series of tasks [13]. These methods tend to perform poorly in class incremental learning [14, 30].

Knowledge Distillation. Knowledge distillation [12] is a widely used method, which transfers key knowledge from a teacher model to a student model. LwF [20] utilizes knowledge distillation to learn multiple tasks. A modified cross-entropy loss is used to preserve the capabilities of old model. Then, it was applied to multi-class classification, called LwF.MC [26]. M^2KD [40] introduces a multi-model and multi-level knowledge distillation strategy, which utilizes all previous model snapshots instead of distilling knowledge only from the last model.

Rehearsal. The rehearsal strategy alleviates catastrophic forgetting by using some old data to make up training data. The simplest approach is to store few old data and replay them in a new incremental step. This straightforward approach has been demonstrated to be effective in many scenarios [14, 30]. Other methods construct a generative model, e.g., GANs [8], to generate samples for rehearsal instead of storing old data directly [6, 29, 34]. However, in these methods, an additional generative model needs to be trained simultaneously. Therefore, they rely heavily on the quality of the generated model.

Class Imbalance. For class incremental learning, data of old classes is generally not available when new classes appear. Even with the rehearsal strategy, the class imbalance problem is still very serious, which is an important factor in catastrophic forgetting [13, 33]. Though class imbalance is an old topic and has attracted a lot of attention [4, 15, 17], multi-class imbalance learning is still an open problem [36]. In order to address it in class incremental learning, BiC [33] adds a bias correction layer to correct the model's outputs. This method needs to keep a validation set to train the additional bias correction layer. In [13], cosine normalization, less-forget constraint, and inter-class separation are incorporated to mitigate the impact of class imbalance. This method combines three specific loss terms and other skills (e.g., class balance fine tune) to improve performance. IL2M [3] rectifies scores of old classes by leveraging contents from a dual memory.

These strategies can be applied in combination. For example, both the distillation strategy and the rehearsal strategy are used in iCaRL [26], which also utilizes a nearest-exemplars-mean (NEM) classifier. EEIL [5] also exploits these two strategies and utilizes a balanced fine tuning to alleviate class imbalance. In this paper, the proposed method is also based on these perspectives. A detailed analysis of distillation strategy is presented. More importantly, we deal with class imbalance in a simple and effective manner. Without any additional model parameters, hyperparameters or a reserved validation set, our method achieves better performance than previous methods.

3. Motivation

3.1. Baseline

In this subsection, we summarize a baseline method in class incremental learning, which utilizes both the rehearsal strategy and the distillation strategy.

Let us first formulate class incremental learning. Assume there are B batches of train data $\{D^1, \cdots, D^B\}$, with $D^b = \{(\mathbf{x}_1^b, y_1^b), \cdots, (\mathbf{x}_{n_b}^b, y_{n_b}^b)\}$ for the b^{th} incremental step, where $\mathbf{x}_i^b$ and y_i^b represent the input data and the target respectively, n_b is the number of samples in the set D^b. In the b^{th} step of class incremental learning, the goal is to learn knowledge from new data D^b, while retain the previous experiences learned from old data $\{D^1, \cdots, D^{b-1}\}$. For each step, the trained model is evaluated on all seen classes.

For the b^{th} incremental step, the baseline method initializes the model with the parameters learned in the previous step and adds new output nodes (weights in the FC layer are initialized randomly). Then, it attempts to learn new classes and meanwhile preserve the original capabilities with the new data D^b and a few rehearsal data D^b_{old}. It is assumed that the new data D^b comes from C^b new classes, and the rehearsal data D^b_{old} comes from C^b_{old} old classes, where $C^b_{old} = \sum_{k=1}^{b-1} C^k$. The baseline method combines the cross-entropy loss $\mathcal{L}_{CE}$ with the knowledge distillation loss $\mathcal{L}_{KD}$. The combined loss containing two terms is given as:

$$\mathcal{L}(\mathbf{x}, y) = (1-\lambda)\mathcal{L}_{CE}(\mathbf{x}, y) + \lambda \mathcal{L}_{KD}(\mathbf{x}), \tag{1}$$

where λ is a hyper-parameter governing the balance between the two losses. We set the hyper-parameter λ to $\frac{C^b_{old}}{C^b + C^b_{old}}$, according to the recommendation in [33]. The cross-entropy loss is given by:

$$\mathcal{L}_{CE}(\mathbf{x}, y) = \sum_{c=1}^{C^b + C^b_{old}} -\delta_{c=y} \log\big(p_c(\mathbf{x})\big), \tag{2}$$

where $\delta_{c=y}$ is the indicator function and $p_c(\mathbf{x})$ is the output probability for the c^{th} class. And the distillation loss is given by:

$$\mathcal{L}_{KD}(\mathbf{x}) = \sum_{c=1}^{C^b_{old}} -\hat{q}_c(\mathbf{x}) \log\big(q_c(\mathbf{x})\big), \tag{3}$$

where $\hat{q}_c(\mathbf{x}) = \frac{e^{\hat{o}_c(\mathbf{x})/T}}{\sum_{j=1}^{C^b_{old}} e^{\hat{o}_j(\mathbf{x})/T}}, q_c(\mathbf{x}) = \frac{e^{o_c(\mathbf{x})/T}}{\sum_{j=1}^{C^b_{old}} e^{o_j(\mathbf{x})/T}}$; T is the temperature scalar; $\hat{o}_c(\mathbf{x})$ is an element of $\hat{\mathbf{o}}(\mathbf{x})$, $\hat{\mathbf{o}}(\mathbf{x}) = \left(\hat{o}_1(\mathbf{x}), \cdots, \hat{o}_{C^b_{old}}(\mathbf{x})\right)^T$, which represents the output logits of the old model obtained in the previous incremental step; $o_c(\mathbf{x})$ is an element of $\mathbf{o}(\mathbf{x})$, $\mathbf{o}(\mathbf{x}) = \left(o_1(\mathbf{x}), \cdots, o_{C^b_{old}}(\mathbf{x}), o_{C^b_{old}+1}(\mathbf{x}), \cdots, o_{C^b_{old}+C^b}(\mathbf{x})\right)^T$, which stands for the output logits of the current model. Note the sample $(\mathbf{x}, y)$ is from both the new data and the rehearsal data. Then, parameters of both the feature extraction layers and the FC layer are updated with the combined loss defined in Eq.(1) during training.

Table 1: Error analysis on two parts of the test set. $e(o)$, $e(n)$ represent the number of old samples and new samples that are wrongly predicted, respectively. Specifically, error analysis for old samples is given in detail: $e(o,n)$, $e(o,o)$ stand for the number of old samples that are misclassified as new classes or other old classes, respectively.

	$e(n)$	$e(o)$	$e(o,n)$	$e(o,o)$
CE	314	5,360	4,027	1,333
CE + KD	383	5,326	4,314	1,012

3.2. Effect of Knowledge Distillation

The baseline method is widely used in class incremental learning. However, there is a lack of explicit analysis of the role of knowledge distillation. To do this, we carry out experiments on the CIFAR-100 [19] with 5 incremental steps ($B = 5$) and 20 classes per step ($C^b = 20, b = 1, \cdots, 5$).

We perform class incremental learning with two methods: (a) using the cross-entropy loss; (b) using both the cross-entropy loss and the distillation loss. After 5 incremental steps, we evaluate the two models trained by method (a) and (b). The test set is comprised of two parts, one containing 80 old classes and another 20 new classes. Error analysis on two parts of the test set is reported in Table 1. There are 2,000 test samples in the new part, and 8,000 samples in the old part. As can be seen, both methods have very poor performance in term of old classes, which shows that they have lost the ability to recognize old data.

We further analyze the type of misclassification of old data. As shown in Table 1, the combined loss reduces the number of old samples that are misclassified to other old classes: 1,012 (CE + KD) vs 1,333 (CE). This is consistent with the original intention of knowledge distillation, that is, to keep the knowledge of old model. However, the prediction bias towards new classes is not alleviated: there are more old samples that are misclassified to new classes: 4,314 (CE + KD) vs 4,027 (CE). Why dose the model trained with the distillation loss become more serious towards new classes? After revisiting the distillation loss, we find the cost of misclassifying old samples to new classes is smaller than that to other old classes. If old samples are misclassified to new classes, the distillation loss

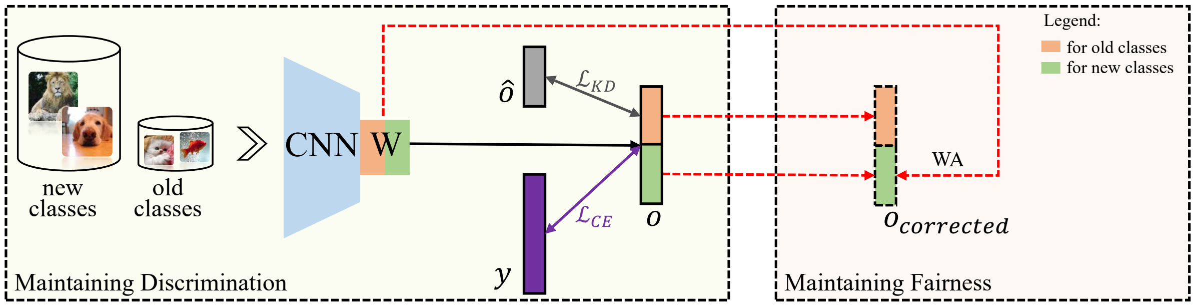


Figure 3: Overview of our solution for class incremental learning. In the first phase, we train the model with the cross-entropy loss ($\mathcal{L}_{CE}$) and the distillation loss ($\mathcal{L}_{KD}$). In the second phase, we correct the biased weights in the trained model via Weight Aligning (WA). $\mathbf{o}$ and $\hat{\mathbf{o}}$ represent the output logits of the current model and the old model respectively, y stands for the true label, $\mathbf{o}_{corrected}$ represents the corrected output logits by using WA.

still can be low, as $\{q_c(\mathbf{x}),\ c = 1, \cdots, C^b_{old}\}$ are only calculated between the outputs corresponding to old classes. While, if they are misclassified to other old classes, the distillation loss will be high, as the output probability distribution is definitely not coincide with the target distribution. As a result, the model is more inclined to misclassify old samples into new classes.

Based on the above analysis, we argue that the positive effect of distillation loss is maintaining the discrimination within old classes, so that it is successful in making fewer misclassifications within old classes. However, the model still has a prediction bias towards new classes. The positive effect of knowledge distillation here is limited. Besides, if there are more than two incremental steps, i.e., $B > 2$, the 'ill' model will become a teacher model in the next incremental step, then the deviation will accumulate, so that the positive effect will be further limited.

4. Methodology

Our method consists of two phases, as shown in Figure 3. The first phase is **Maintaining Discrimination**. In this phase, we train a new model on the new data and the rehearsal data with the combined loss. We expect to transfer knowledge from the old model to the new model and maintain discrimination within old classes with the help of knowledge distillation.

As knowledge distillation loss still cannot help the model to treat old classes and new classes fairly as shown in subsection 3.2, we design the second phase, called **Maintaining Fairness**. In this phase, we propose a method named Weight Aligning (WA) to correct the model trained in the first phase. The corrected model treats old classes and new classes fairly, and can significantly improve the overall performance.

4.1. Biased Weights in the FC Layer

As shown in subsection 3.2, the model trained via the baseline method still tends to predict test samples as new classes. To study this problem conveniently, we express the FC layer of model in the b^{th} incremental step in the following form:

$$\mathbf{o}(\mathbf{x}) = \mathbf{W}^T \phi(\mathbf{x}), \tag{4}$$

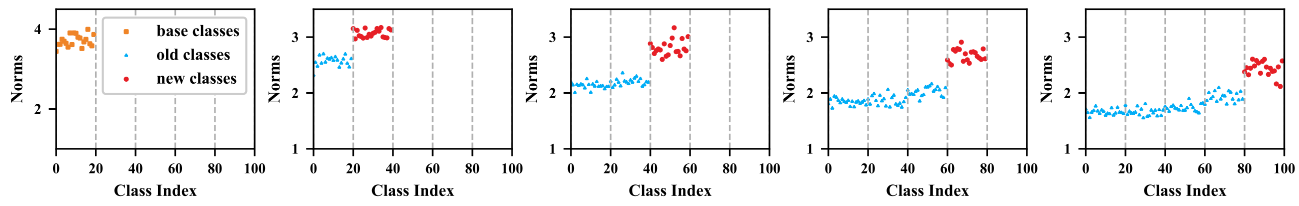
where the $(C^b_{old} + C^b)$-dimensional vector $\mathbf{o}(\mathbf{x})$ represents output logits of the current model; $\phi(\cdot)$ is a feature extraction function (can be a CNN-based model usually), which outputs d-dimensional feature vectors; $\mathbf{W} \in \mathbb{R}^{d\times(C^b_{old}+C^b)}$ stands for the weights, which can be expressed as $\mathbf{W} = \{\mathbf{w}_c, 1 \le c \le C^b_{old} + C^b\}$, where $\mathbf{w}_c$ is a d-dimensional weight vector for the c^{th} class. Note, for the convenience of analysis, we always set the bias term in the FC layer to zero without special instructions, which will be discussed in the ablation study.

We carry out experiments on CIFAR-100 with 5 incremental steps and 20 classes per step. After each step, we calculate the norms of the weight vectors $\{\mathbf{w}_c\}$ and plot them in Figure 4. As shown in Figure 4 (b), (c), (d) and (e), the norms of the weight vectors for new classes are much larger than those for old classes. This phenomenon is mainly caused by class imbalance [9, 21]. Due to the output logits for the c^{th} class is calculated as

$$o_c(\mathbf{x}) = \mathbf{w}_c^T \phi(\mathbf{x}), \tag{5}$$

if the norms of weight vectors for new classes are larger, the output logits for new classes may tend to be larger in general. As a result, the trained model may tend to predict an input image as belonging to a new class.

However, as shown in Figure 4 (a), in the first phase, the norms of the weight vectors are roughly equal, as this phase does not related to class incremental learning actually. We treat this as a priori knowledge. The phenomenon in class incremental learning does not match this prior knowledge, which inspires us to correct the biased weights.



(a) $C^1 = 20, C^1_{old} = 0$ (b) $C^2 = 20, C^2_{old} = 20$ (c) $C^3 = 20, C^3_{old} = 40$ (d) $C^4 = 20, C^4_{old} = 60$ (e) $C^5 = 20, C^5_{old} = 80$

Figure 4: Norms of the weight vectors $\{\mathbf{w}_c\}$. (a) Results of the 1^{st} step (20 base classes), which does not correspond to class incremental learning; (b), (c), (d) and (e) are the results of the 2^{nd}, 3^{rd}, 4^{th}, 5^{th} incremental step respectively, which show the norms of the weight vectors of new classes are much larger than those of old classes. (Best viewed in color)

4.2. Weight Aligning

Based on the above, we present a simple and effective approach, called Weight Aligning (WA), to correct the biased weights in the FC layer. In WA, the norms of the weight vectors of new classes are aligned to those of old classes.

Firstly, we rewrite the weights in the FC layer in the following form

$$\mathbf{W} = (\mathbf{W}_{old}, \mathbf{W}_{new}),$$

where

$$\mathbf{W}_{old} = (\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_{C^b_{old}}) \in \mathbb{R}^{d \times C^b_{old}},$$
$$\mathbf{W}_{new} = (\mathbf{w}_{C^b_{old}+1}, \cdots, \mathbf{w}_{C^b_{old}+C^b}) \in \mathbb{R}^{d \times C^b}.$$

Then, we denote, respectively, the norms of the weight vectors of old classes and new classes as follows

$$\boldsymbol{Norm}_{old} = (||\mathbf{w}_1||, \cdots, ||\mathbf{w}_{C^b_{old}}||),$$
$$\boldsymbol{Norm}_{new} = (||\mathbf{w}_{C^b_{old}+1}||, \cdots, ||\mathbf{w}_{C^b_{old}+C^b}||).$$

Based on the above norms, we normalize the weights for new classes by

$$\widehat{\mathbf{W}}_{new} = \gamma \cdot \mathbf{W}_{new}, \tag{6}$$

where

$$\gamma = \frac{Mean(\boldsymbol{Norm}_{old})}{Mean(\boldsymbol{Norm}_{new})}, \tag{7}$$

$Mean(\cdot)$ returns the mean value of elements in the vector. In this way, the average norm of the weight vectors for new classes becomes the same as that for old classes. Note that we only make the average norms become equal, in other words, within new classes (or old classes), the relative magnitude of the norms of the weight vectors does not change. Such a design is mainly used to ensure the data within new classes (or old classes) can be separated well.

The original output logits of the model trained in the first phase of our method can be expressed as

$$\mathbf{o}(\mathbf{x}) = \begin{pmatrix} \mathbf{o}_{old}(\mathbf{x}) \\ \mathbf{o}_{new}(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \mathbf{W}^T_{old}\ \phi(\mathbf{x}) \\ \mathbf{W}^T_{new}\ \phi(\mathbf{x}) \end{pmatrix} \tag{8}$$

After applying WA to the weights, the corrected output logits are given by:

$$\begin{aligned} \mathbf{o}_{corrected}(\mathbf{x}) &= \begin{pmatrix} \mathbf{W}^T_{old}\ \phi(\mathbf{x}) \\ \widehat{\mathbf{W}}^T_{new}\ \phi(\mathbf{x}) \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{W}^T_{old}\ \phi(\mathbf{x}) \\ \gamma \cdot \mathbf{W}^T_{new}\ \phi(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \mathbf{o}_{old}(\mathbf{x}) \\ \gamma \cdot \mathbf{o}_{new}(\mathbf{x}) \end{pmatrix} \end{aligned} \tag{9}$$

As shown in Eq.(9) and Eq.(7), the final effect of aligning the weights is to rescale the output logits of new classes by a coefficient. The latter experiments demonstrate that our method can effectively alleviate the prediction bias.

4.3. Restriction to the Weights

In fact, the magnitude relationship between the norms of weight vectors for new classes and those for old classes may not always reflect the magnitude relationship between the output logits for old classes and those for new classes. Suppose that the feature extraction function provides the feature vectors, whose elements are all non-negative. This assumption is reasonable, because in usual model architectures, the learned features are activated by the 'ReLU' function $\big(\text{ReLU}(x) = \max(0, x)\big)$, which returns non-negative values. As the weight vectors $\{\mathbf{w}_c\}$ usually contain both positive and negative elements, the negative elements with large absolute values contribute to a large norm of weight vectors. However, they are not in favor of large output logits. Thus, in order to make the norm of the weight vector $\mathbf{w}_c$ more consistent with its corresponding output logits, we restrict the elements of the weight vector $\mathbf{w}_c$ to be positive. To achieve this, weight clipping [2] can be performed after each optimization step in training. The impact of restricting the weights in the FC layer to be positive will be analyzed in the ablation study.

5. Experiments

5.1. Experimental Settings

We evaluate the methods on ImageNet ILSVRC 2012 [27] and CIFAR-100 [19], which are widely used in the

Table 2: Class incremental learning performance (top-1 accuracy %) on CIFAR-100 with 5 incremental steps and 20 classes per step. The gains on the basis of Variation1 are also reported in parentheses. 'Full' is obtained with all training data for all classes. The average results over all the incremental steps except the first step are also reported here.

#classes	20	40		60		80		100		Average	
Variation1 (CE)	83.5	70.7		58.2		49.2		43.3		55.3	
Variation2 (CE + WA)	83.5	74.3	(+3.6)	64.0	(+5.8)	56.9	(+7.7)	50.8	(+7.5)	61.5	(+6.2)
Variation3 (CE + KD)	83.5	72.8	(+2.1)	60.1	(+1.9)	49.9	(+0.7)	42.9	(-0.4)	56.4	(+1.1)
Variation4 (CE + KD + WNL)	83.1	72.3	(+1.6)	61.6	(+3.4)	53.1	(+3.9)	46.0	(+2.7)	58.2	(+2.9)
Ours (CE + KD + WA)	83.5	**75.5**	(+4.8)	**68.7**	(+10.5)	**63.1**	(+13.9)	**59.2**	(+15.9)	**66.6**	(+11.3)
Full	–							70.1		–	

study of class incremental learning [5, 26, 33]. ImageNet ILSVRC 2012 is a large-scale dataset with 1,000 classes that includes about 1.2 million images for training and 50,000 images for validation. CIFAR-100 consists 32×32 pixel color images with 100 classes. It contains 50,000 images for training with 500 images per class, and 10,000 images for evaluating with 100 images per class.

Our method are implemented with Pytorch [24]. The code will be made publicly available. For ImageNet, we adopt a 18-layer ResNet [10, 11]. We use SGD to train our model and set the batch size to 256. The learning rate starts from 0.1 and reduces to 1/10 of the previous learning rate after 30, 60, 80 and 90 epochs (100 epochs in total). For CIFAR-100, we use a 32-layer ResNet. We also train the model with SGD and set the batch size to 32. The learning rate starts from 0.1 and reduces to 1/10 of the previous learning rate after 100, 150 and 200 epochs (250 epochs in total). We set the temperature scalar T to 2. For data augmentation, random cropping, horizontal flip and normalization are employed to augment training images.

5.2. Effect of Weight Aligning

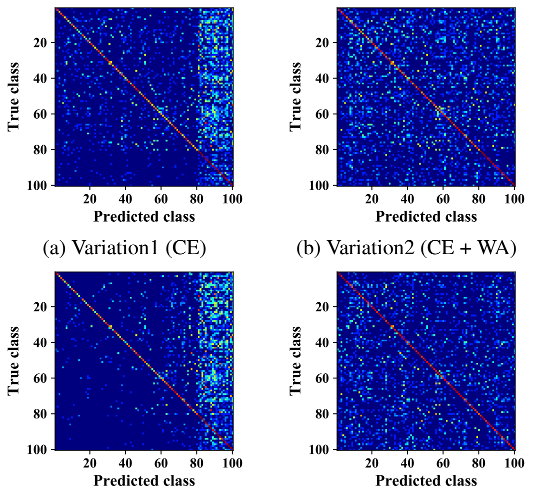
To analyze the effect of weight aligning, we perform experiments on CIFAR-100 with 5 incremental steps and 20 classes per step. We first compare our method with three variations in the following: **Variation1**, training with the cross-entropy loss; **Variation2**, training with the cross-entropy loss, and correcting the model via WA; **Variation3**, training with the combined loss; **Ours**, training with the combined loss and correcting the model via WA.

Table 2 summarizes the results of these experiments. Variation1 is the worst one, as it only uses the cross-entropy loss. Variation3 adds the distillation loss on the basis of Variation1 to mitigate catastrophic forgetting. However, Variation3 is only a little better than Variation1. Variation2 uses WA to correct the model based on Variation1, and significantly improves performance (the gain in term of the overall performance at the end of class incremental learning is 7.5%). From the results of 'Ours', WA also gets significant improvements (more than 16% at the end of class incremental learning over Variation3). These results demonstrate that WA is quite effective for class incremental learning.

It is worth noting that the gain brought by the combination of KD and WA is greater than the sum of the gains from each component used separately, e.g., for the average results, the gain of the combination (Ours) is 11.3%, and the gains of WA (Variation2) and KD (Variation3) used separately are 6.2% and 1.1% respectively. As shown in subsection 3.2, the positive effect of KD is limited when used alone. KD helps the model to output more discriminative results within old classes, however, these outputs are overwhelmed by the superior outputs of new classes. For example, as shown in Figure 2, with the help of KD, the output probability for 'cat' becomes higher than that for 'fish', but still lower than that for new class 'lion' or 'dog'. In such a scenario, the positive effect of KD is hidden. As our method maintains not only the discrimination within old classes but also the fairness between old classes and new classes, it strengthens the positive effect of KD. On the other hand, the corrected outputs via WA are more accurate with the help of KD. Therefore, our method creates the "one plus one greater than two" effect and achieves significant improvements.

The confusion matrices of different methods are presented in Figure 5. From Figure 5 (a) and (c), we see that KD leads to fewer misclassifications between old classes, however, both Variation1 and Variation3 tend to predict objects as new classes. With the help of WA, Variation2 and our method make the model treat new classes and old classes fairly as shown in Figure 5 (b) and (d). And our method achieves better performance with the help of KD. These results intuitively show that the proposed method can effectively maintain discrimination and fairness in the model predictions.

The proposed method weight aligning is a post-processing technique. It is interesting to see the effect of adding a normalization layer on the weights (in the FC layer) directly, like the operation in Modified Softmax Loss [21] and NormFace [31], so that the weights of all classes can have a unit norm. We implement this method as **Varia-**



(a) Variation1 (CE) (b) Variation2 (CE + WA) (c) Variation3 (CE + KD) (d) Ours (CE + KD + WA)

Figure 5: Confusion matrices of different approaches.

Table 3: Class incremental learning performance (top-5 accuracy %) on ImageNet (1,000 classes and 100 classes) with 10 incremental steps. The performance at the last incremental step and the average results over all the incremental steps except the first step are reported here. The results of the compared methods are reported in the original papers.

#classes	1000		100	
	Last	Average	Last	Average
LwF.MC [20, 26]	24.3	42.5	36.6	60.7
iCaRL [26]	44.0	60.8	63.8	81.8
EEIL [5]	52.3	69.4	80.2	89.2
BiC [33]	73.2	82.9	**84.4**	89.8
IL2M [3]	–	78.3	–	–
RPS [25]	–	–	74.0	86.6
Ours	**81.1**	**85.7**	84.1	**90.2**
Full	89.1	–	95.1	–

tion4: training with the combined loss and a weight normalization layer (WNL). The results are also provided in Table 2. Compared with Variation1 and Variation2, this method does not bring about a significant improvement. Actually, the FC layer plays an important role in the visual representation transfer [37]. If the weights in the FC layer are strictly limited during the training process, in order to adapt to new data, the bias in the feature extraction layers will become more serious. However, the bias in the feature extraction layers is harder to correct than that in the weights of FC layer, as the parameters of feature extraction layers are shared by all classes and the weights of FC layer are not shared between classes. Therefore, it is better to take a post-processing approach, such as WA. In addition, we have tested the method that normalizing the weights of all classes to have a unit norm after the usual training process. While this approach is inferior to WA. As mentioned in subsection 4.2, within the new classes (or the old classes), the relative magnitude of the norms of the weight vectors does not change in WA, such a design can maintain the differences and ensure that the classes can be separated well.

5.3. Comparison to Other Methods

We compare our method with several competitive or representative methods, including LwF.MC [20, 26], iCaRL [26], EEIL [5], BiC [33], IL2M [3], RPS [25]. Experiments are performed on ImageNet and CIFAR100.

Evaluation on ImageNet. We conduct two experiments on this dataset. In the first one, 100 classes (ImageNet-100) are selected randomly and split into 10 incremental batches with 10 classes per batch; In the second one, we split the 1000 classes (ImageNet-1000) into 10 incremental batches with 100 classes per batch. For the sake of fairness, we use the same set of classes in ImageNet-100 and ImageNet-1000 as the previous work [33]. We store 2,000 images for old classes in ImageNet-100 experiments. And in ImageNet-1000 experiments, we store 20,000 images for old classes as the same as the previous work. We select rehearsal exemplars based on herding selection [32] which is also the same as the previous work. More classes have been seen, fewer images can be retained per class. As a result, the problem of class imbalance becomes more serious.

The class incremental learning results (top-5 accuracy %) on ImageNet-100 and Imagenet-1000 are shown in Table 3. We report the performance at the last incremental step and the average results over all the incremental steps except the first step here (as the first step does not related to class incremental learning actually). We also provide the detailed results of all incremental steps and the top-1 results in the supplementary material. As can be seen from these tables, the proposed method outperforms the compared methods by a large margin, especially on the large scale dataset ImageNet-1000. The overall performance at the end of class incremental learning is improved by more than 28% compared to EEIL on ImageNet-1000. In contrast to the state-of-the-art method BiC, the proposed method also achieves better results (surpasses it by 7.9% at the end of class incremental learning on ImageNet-1000). Though Eq.(9) is similar in form to the linear model in BiC, the proposed method does not need to reserve a validation set which is used in BiC to learn additional parameters. All of the rehearsal data can be utilized to learn a better feature extractor, so that the proposed method can outperform BiC.

Overall, these results indicate that the proposed method is effective to handle catastrophic forgetting in class incremental learning. Our approach not only achieves better per-

Table 4: Class incremental learning performance (top-1 accuracy %) on CIFAR100 with 2, 5, 10 and 20 incremental steps. The average results over all the incremental steps except the first step are reported.

#incremental steps	2	5	10	20
LwF.MC [20, 26]	52.6	47.1	39.7	29.7
iCaRL [26]	62.0	63.3	61.6	59.7
EEIL [5]	60.8	63.7	63.6	**63.4**
BiC [33]	64.9	65.1	63.5	62.1
Ours	**65.1**	**66.6**	**64.5**	62.6
Full	70.1			

formance but also has a simpler structure.

Evaluation on CIFAR-100. CIFAR-100 has 100 classes, which are divided into 2, 5, 10 and 20 incremental batches respectively in our experiments. The same set of classes in CIFAR-100 are used for all of the compared methods. In CIFAR-100 experiments, we store 2,000 samples in total as the same as previous work.

The average results over all the incremental steps except the first step are shown in Table 4. Detailed results of all incremental steps are reported in the supplementary material. On CIFAR-100, these methods achieve similar results, which is mainly because this dataset is simple [33]. Consistent with the results on ImageNet, the proposed method achieves better results compared to state-of-the-art approaches on CIFAR-100 under different settings.

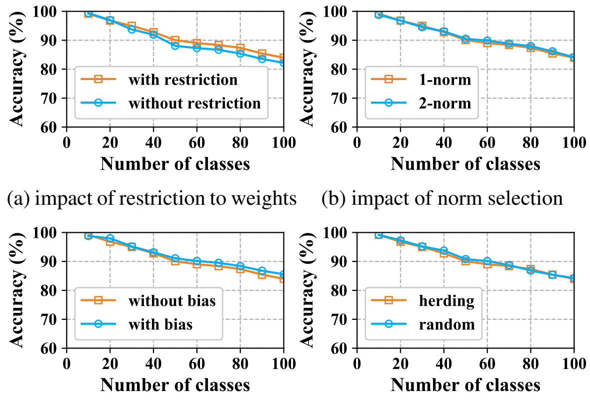
5.4. Ablation Study

In this subsection, we analyze the impact of the components of our method. More analysis can be found in the supplementary material.

Impact of Restriction to the Weights. We studied the impact of restricting the weights in the FC layer to be positive on ImageNet-100 with 10 incremental steps. As shown in Figure 6 (a), our method obtained better performance with restriction to the weights. As discussed in subsection 4.3, this is mainly due to the norms of the weight vectors become more consistent with their corresponding output logits when restricting the weights to be positive, so that the scale factor γ obtained by Eq.(7) is more accurate to suppress the output logits of new classes.

Impact of Norm Selection. We investigated the impact of different norm used in the proposed method. We compare two norms: 1-norm and 2-norm. Figure 6 (b) shows the results. 1-norm and 2-norm achieve similar results, which indicates our method is not sensitive to norm selection.

Impact of the Bias Term in the FC Layer. We studied the impact of the bias term. With the bias term, the proposed method still calculates the scale factor γ by Eq.(7) based on the weight information and applies it to the output logits for new classes. In other words, the scalar factor γ obtained from weight information is used in both the weight term and the bias term in the FC layer. We compare our method with or without using the bias term in the FC layer. Figure 6 (c) shows the results. We see that the bias term in the FC layer can only influence the performance slightly.



(a) impact of restriction to weights (b) impact of norm selection

(c) impact of the bias term (d) impact of exemplar selection

Figure 6: Class incremental learning performance (top-5 accuracy %) on ImageNet-100 for ablation study.

Impact of Exemplar Selection Strategies. We investigated the impact of exemplar selection strategies. Random selection and herding selection are considered. Figure 6 (d) shows the results. We see that the exemplar selection strategies can only influence the performance slightly.

6. Conclusions

The goal of class incremental learning is to obtain desirable results on new data, at the same time, retain the previous learned experiences. In this paper, we investigated catastrophic forgetting in class incremental learning. We demonstrated the actual role of knowledge distillation in this problem and the heavily biased weights in the FC layer. We proposed a simple and effective solution to address catastrophic forgetting that maintains the discrimination via knowledge distillation and maintains the fairness via a method called weight aligning. The experimental results on ImageNet-1000, ImageNet-100, and CIFAR-100 show that the proposed method achieves better performance than the previous methods. This work may suggest that there are many useful information hidden in the trained model that is worth exploring.

Acknowledgements This work is supported in part by the National Natural Science Foundation of Guangdong Province (2018A030313422), the National Natural Science Foundation of China (61972219, 61771273), the National Key Research and Development Program of China (2018YFB1800204, 2018YFB1800600), the Key Research and Development Program for Guangdong Province (2019B010136001), the Science and Technology Planning Project of Guangdong Province (LZC0023, LZC0024), the R&D Program of Shenzhen (JCYJ20190813174403598, JCYJ20180508152204044), and the research fund of PCL Future Regional Network Facilities for Large-scale Experiments and Applications (PCL2018KP001).

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018. 1, 2

[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *ArXiv*, abs/1701.07875, 2017. 5

[3] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 7

[4] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks : the official journal of the International Neural Network Society*, 106:249–259, 2018. 2

[5] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 233–248, 2018. 1, 2, 6, 7, 8

[6] Sebastian Farquhar and Yarin Gal. A unifying bayesian view of continual learning. *arXiv preprint arXiv:1902.06494*, 2019. 2

[7] Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3:128–135, 1999. 1, 2

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[9] Yandong Guo and Lei Zhang. One-shot face recognition by promoting underrepresented classes. *ArXiv*, abs/1707.05574, 2017. 4

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. 1, 6

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *ArXiv*, abs/1603.05027, 2016. 1, 6

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 2

[13] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 1, 2

[14] Yen-Chang Hsu, Yen-Cheng Liu, and Zsolt Kira. Reevaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018. 2

[15] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5375–5384, 2016. 2

[16] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2016. 1

[17] Salman Hameed Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous Ahmed Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, 29:3573–3587, 2015. 2

[18] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 1, 2

[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 3, 5

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 2, 7, 8

[21] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Min Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6738–6746, 2017. 4, 6

[22] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165, 1989. 1, 2

[23] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. *ArXiv*, abs/1710.10628, 2017. 1

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. 6

[25] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for incremental learning. *ArXiv*, abs/1906.01120, 2019. 7

[26] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 1, 2, 6, 7, 8

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015. 5

[28] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 1

[29] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pages 2990–2999, 2017. 1, 2

[30] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. *CoRR*, abs/1904.07734, 2019. 2

[31] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. *ArXiv*, abs/1704.06369, 2017. 6

[32] Max Welling. Herding dynamical weights to learn. In *ICML*, 2009. 7

[33] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 1, 2, 3, 6, 7, 8

[34] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Fu. Incremental classifier learning with generative adversarial networks. *ArXiv*, abs/1802.00853, 2018. 1, 2

[35] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3987–3995. JMLR. org, 2017. 1, 2

[36] Chongsheng Zhang, Jingjun Bi, Shixin Xu, Enislay Ramentol, Gaojuan Fan, Baojun Qiao, and Hamido Fujita. Multi-imbalance: An open-source software for multi-class imbalance learning. *Knowl.-Based Syst.*, 174:137–143, 2019. 2

[37] Chen-Lin Zhang, Jian-Hao Luo, Xiu-Shen Wei, and Jianxin Wu. In defense of fully connected layers in visual representation transfer. In *PCM*, 2017. 7

[38] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry P. Heck, Heming Zhang, and C.-C. Jay Kuo. Class-incremental learning via deep model consolidation. *ArXiv*, abs/1903.07864, 2019. 2

[39] Xu Zhang, Yang Yao, Baile Xu, Lekun Mao, Shen Furao, Jian Zhao, and Qingwei Lin. Label mapping neural networks with response consolidation for class incremental learning. *ArXiv*, abs/1905.07835, 2019. 1

[40] Peng Zhou, Long Mai, Jianming Zhang, Ning Xu, Zuxuan Wu, and Larry S Davis. M2kd: Multi-model and multi-level knowledge distillation for incremental learning. *arXiv preprint arXiv:1904.01769*, 2019. 1, 2