GKEAL: Gaussian Kernel Embedded Analytic Learning for Few-shot Class Incremental Task

Huiping Zhuang[1*], **Zhenyu Weng**[2], **Run He**[1], **Zhiping Lin**[2], **Ziqian Zeng**[1]
[1]Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, China
[2]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[*]corresponding: hpzhuang@scut.edu.cn

Abstract

Few-shot class incremental learning (FSCIL) aims to address catastrophic forgetting during class incremental learning in a few-shot learning setting. In this paper, we approach the FSCIL by adopting analytic learning, a technique that converts network training into linear problems. This is inspired by the fact that the recursive implementation (batch-by-batch learning) of analytic learning gives identical weights to that produced by training on the entire dataset at once. The recursive implementation and the weight-identical property highly resemble the FSCIL setting (phase-by-phase learning) and its goal of avoiding catastrophic forgetting. By bridging the FSCIL with the analytic learning, we propose a Gaussian kernel embedded analytic learning (GKEAL) for FSCIL. The key components of GKEAL include the kernel analytic module which allows the GKEAL to conduct FSCIL in a recursive manner, and the augmented feature concatenation module that balances the preference between old and new tasks especially effectively under the few-shot setting. Our experiments show that the GKEAL gives state-of-the-art performance on several benchmark datasets.

1. Introduction

Class-incremental learning (CIL) [20] can continuously absorbs new category knowledge in a phase-by-phase manner with data coming separately in each phase, after training a classification network. This is important as data can be scattered at various times and locations in a non-identical independent way. The few-shot class incremental learning (FSCIL) [23] further imposes an inefficiency constraint on the data availability. That is, only a few data samples, i.e., few-shot, for each new class is allowed, leading to a more challenging incremental learning problem.

The major challenge for FSCIL follows from the CIL's, namely the *catastrophic forgetting*. The performance on old (base) tasks is tremendously discounted after learning new tasks. This is caused by the lack of training data for old tasks, tricking models to focus only on new tasks. The forgetting issue is also referred to as *task-recency* bias, in favor of newly learned tasks in prediction. The forgetting issue in FSCIL manifests more quickly due to over-fitting than that in the conventional CIL setting as the training samples become scarce for new tasks.

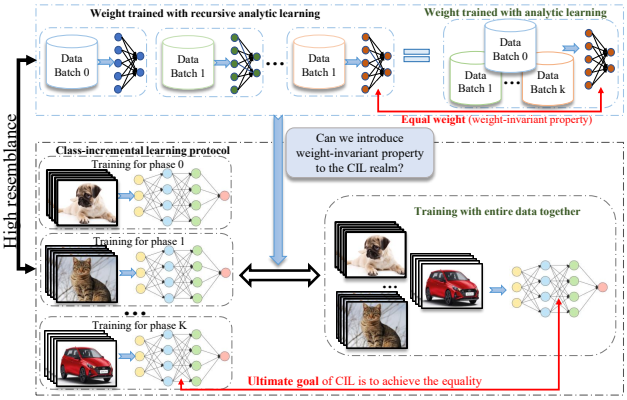

Figure 1. The resemblance between the analytic learning (recursive form) [33] and incremental learning. We want to build a bridge between these two fields to take advantage of the analytic learning for addressing the FSCIL.

To handle the forgetting, conventional CIL sparks various contributions, which mainly include the *Bias correction-based* CIL [1, 9], *Regularization-based* CIL [11, 13] and *Replay-based* CIL [17, 20]. They work well in addressing the catastrophic forgetting in CIL. However, the few-shot constraint in FSCIL renders the CIL solutions obsolete (see [23] or our experiments). There have been several works [22, 23, 31] taking into account the few-shot constraint, outperforming the conventional CIL. These FSCIL techniques take inspirations from existing CIL variants [22] or the few-shot learning angle (e.g., prototype-based [31]) to present catastrophic forgetting.

In this paper, inspired by *analytic learning* [33, 34]—a technique converting network training into linear problems—we approach the FSCIL in an unique angle by incorporating traditional machine learning techniques. The analytic learning allows the training to be implemented in a recursive manner where training data are scattered into multiple batches. Yet the weights trained recursively are identical to those trained by pouring the entire data in one go [33]. We may call this *weight-invariant* (or weight-identical) property. Such recursive form and its weight-invariant property highly resemble the incremental learning paradigm and its objective of avoiding (catastrophic) forgetting respectively (see Figure 1). Following this intuition, we propose a Gaussian kernel embedded analytic learning (GKEAL) for FSCIL. The GKEAL adopts traditional machine learning tools such as least squares (LS) and matrix inverse to avoid forgetting. The key contributions are summarized as follows.

• We introduce GKEAL by treating the FSCIL as a recursive learning problem to avoid forgetting. We prove that the GKEAL in the FSCIL setting follows the same weight-invariant property as that in analytic learning.

• To bridge analytic learning into the FSCIL realm, the GKEAL replaces the classifier at a network's final layer with a kernel analytic module (KAM). The KAM contains a Gaussian kernel embedding process for extracting more discriminative feature, and an LS solution that allows the GKEAL to learn new tasks in a recursive manner.

• To mitigate the data imbalance between the base and new tasks, an augmented feature concatenation (AFC) module is introduced, which effectively balances the network's base-new task preference.

• Experiments on benchmark datasets show that the GKEAL outperforms the state-of-the-art methods by a considerable margin. Ablation study is also provided, giving thorough analysis of the hyperparameters introduced, as well as strong supports to our theoretical claims.

2. Related Works

Class-Incremental Learning. Existing CIL methods mainly include three categories, namely Bias correction-based CIL, Regularization-based CIL and Replay-based CIL. The bias correction-based CIL aims to address the task-recency bias. The end-to-end incremental learning (EEIL) [1] introduces a task balance stage in order to reduce the bias. In [27], the bias is corrected by introducing an additional trainable layer. The method named LUCIR [9] replaces the softmax layer with a cosine normalization alternative to reduce target-recency bias.

The regularization-based CIL imposes certain constraints on the objective functions in order to prevent forgetting. In [11], the elastic weight consolidation (EWC) estimates the importance with a Fisher information matrix and constrains those weights. The EWC is later enhanced by [15] which seeks a better Fisher matrix approximation. In [13], the learning without forgetting method refrains the activations of old tasks from changing too much while absorbing new tasks.

The relay-based CIL has recently become the favored CIL branch due to its competitive performance to resist the catastrophic forgetting by allowing a small mount of historical data. The incremental classifier and representation Learning (iCaRL) [20] introduced such a setting. Following the iCaRL, the PODNet proposed in [4] employs a spatial-based distillation component, achieving relatively outstanding results especially for large-phase CIL. The AANets [17] balances the stability and plasticity with a stable block and a plastic block. The reinforcement memory management [18] handles the forgetting issue adopting reinforcement learning, and plugging it into PODNet and AANets leads to better performance.

Few-shot Learning. The few-shot learning (FSL) addresses the scenario where each category/task is given only a few training samples. Existing FSL methods are mainly optimization-based [5, 10, 16] and metric-based [6, 28, 29]. Optimization techniques allow a fast adaptation to new few-shot tasks by learning an optimization algorithm. Metric-based methods alleviate distance metrics, e.g., DeepEMD [30], in order to measure the deviation between samples.

Few-shot Class-Incremental Learning. The FSCIL [2, 23, 32] jointly incorporates the settings of CIL and FSL by performing CIL tasks with each phase containing limited samples for new tasks. The topology-preserving knowledge Incrementer (TOPIC) framework [23] mitigates the forgetting issue by stabilizing a neural gas network't topology. In [31], the continually evolved classifier (CEC) separates each class with an independent classifier, and adopts a graph model to propagate context information between classifiers. The F2M [22] overcomes the catastrophic forgetting via finding flat minima. This is achieved by injecting noise during base training, suggesting to take the focus of FSCIL back in the base training stage.

Analytic Learning. The analytic learning is developed to avoid the limitations imposed by back-propagation (BP) so that the training can be completed within one epoch. It is also know as *pseudoinverse learning* [7] due to adopting matrix inverse. The analytic learning begins in the shallow learning. For instance, the radial basis network [19] trains the parameters using an LS estimation in the final layer after conducting a kernel transformation in the first layer. Analytic learning with multiple layers [24, 26, 34] usually converts the nonlinear layers into linear segments, so that LS solutions can be employed in a one-epoch training style. To address the memory issue in analytic learning, the block-wise recursive Moore-Penrose learning (BRMP) [33] is developed, allowing analytic learning to stream new samples

without forgetting the impact of previous learned knowledge. This coincides well with the incremental scenarios and it is introduced to CIL realm in the analytic class-incremental learning (ACIL) [36] which has state-of-the-art performance via a frozen backbone and linear classifier trained by analytic learning. These become the key inspirations for the proposed GKEAL.

3. The Proposed Method

In this section, algorithmic details of the proposed GKEAL are provided. Firstly, the GKEAL adopts a classifier re-training phase on the base task, in which the last-layer classifier is replaced by the KAM. Subsequently the FSCIL tasks are conducted in a recursive manner. The FSCIL discussed in this paper is restricted to classification problems. An overview of GKEAL is depicted in Figure 2.

3.1. Base Training

Base Training via BP. Prior to FSCIL, the network is first trained with BP on the base task (see Figure 2(a)). Here we discuss a commonly seen CNN structure consisting a stack of CNN layers (or known as CNN backbone) as feature extractor followed by a fully-connected network (FCN) as classifier. Let $\boldsymbol{W}_{\text{CNN}}$ and $\boldsymbol{W}_{\text{FCN}}$ represent the weights for the CNN backbone and the FCN classifier. Given an input $\boldsymbol{X} \in \mathbb{R}^{w\times h\times 3}$ (e.g., color image as an example), the output of the network is

$$\boldsymbol{Y} = f_{\text{softmax}}(f_{\text{flat}}(f_{\text{CNN}}(\boldsymbol{X}, \boldsymbol{W}_{\text{CNN}}))\boldsymbol{W}_{\text{FCN}})$$

where $f_{\text{CNN}}(\boldsymbol{X}, \boldsymbol{W}_{\text{CNN}})$ indicates the output of the CNN backbone after feeding the input $\boldsymbol{X}$; f_{flat} is a flattening operator, which flattens an $m-$D tensor into a 1-D vector; f_{softmax} is the softmax function.

Analytic Initialization. With the network trained on base dataset, the GKEAL seeks to detach the CNN backbone, and attach it with the KAM (see Figure 2(b)). Specifically, the KAM is a 2-layer shallow network with the first layer conducting a kernel embedding for further feature extraction, and the second layer being a linear FCN layer. For convenience, the re-training can be referred to as *Analytic Initialization* (AInit).

Prior to more developments, some definitions related to FSCIL must be presented. Suppose that the FSCIL asks for a K-phase learning with each phase (e.g., phase k) given training data $\mathcal{D}_k^{\text{train}} \sim \{\boldsymbol{X}_k^{\text{train}}, \boldsymbol{Y}_k^{\text{train}}\}$ $(k = 1, \ldots, K)$ of disjoint classes. $\boldsymbol{X}_k^{\text{train}} \in \mathbb{R}^{N_k\times w\times h\times 3}$ (e.g., images with a shape of $w \times h \times 3$) and $\boldsymbol{Y}_k^{\text{train}} \in \mathbb{R}^{N_k\times d_{y_k}}$ (with phase k including d_{y_k} classes) are stacked N_k-sample input and label (one-hot) tensors. The objective of FSCIL at phase k is to train networks given $\mathcal{D}_k^{\text{train}}$ and test them on $\mathcal{D}_{0:k}^{\text{test}}$ (with $\mathcal{D}_k^{\text{test}} \sim \{\boldsymbol{X}_k^{\text{test}}, \boldsymbol{Y}_k^{\text{test}}\}$) consisting of all the seen classes in $\mathcal{D}_{0:k}^{\text{train}}$. Specifically, $\mathcal{D}_0^{\text{train}} \sim \{\boldsymbol{X}_0^{\text{train}}, \boldsymbol{Y}_0^{\text{train}}\}$ represents the base training set.

The first step of AInit is to obtain embedding from the detached CNN backbone, shown as follows.

$$\boldsymbol{X}_0^{(\text{cnn})} = f_{\text{flat}}(f_{\text{CNN}}(\boldsymbol{X}_0^{\text{train}}, \boldsymbol{W}_{\text{CNN}})) \tag{1}$$

where $\boldsymbol{X}_0^{(\text{cnn})} \in \mathbb{R}^{N_0\times d_{\text{cnn}}}$ is the embedding of $\boldsymbol{X}_0^{\text{train}}$. Subsequently, we conduct a *Gaussian kernel embedding* (GKE) process to obtain kernelized embedding $\boldsymbol{X}_0^{(\text{ke})}$, i.e.,

$$\boldsymbol{X}_0^{(\text{ke})} = g_{\{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_I\}}(\boldsymbol{X}_0^{(\text{cnn})}) \tag{2}$$

where g indicates the GKE module. The j^{th} row of $\boldsymbol{X}_0^{(\text{ke})}$ writes

$$\begin{aligned}\boldsymbol{X}_0^{(\text{ke})}[j,:] = [&\mathrm{e}^{-\beta\left\|\boldsymbol{X}_0^{(\text{cnn})}[j,:]-\boldsymbol{c}_1\right\|^2} \ldots \\ &\mathrm{e}^{-\beta\left\|\boldsymbol{X}_0^{(\text{cnn})}[j,:]-\boldsymbol{c}_i\right\|^2} \ldots \mathrm{e}^{-\beta\left\|\boldsymbol{X}_0^{(\text{cnn})}[j,:]-\boldsymbol{c}_I\right\|^2}]\end{aligned} \tag{3}$$

where $\{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_I\}$ is a set of center vectors randomly selected from the rows of $\boldsymbol{X}_0^{(\text{cnn})}$, and β is a width-adjusting parameter. Next, the kernelized embedding $\boldsymbol{X}_0^{(\text{ke})}$ is mapped onto the label matrix $\boldsymbol{Y}_0^{\text{train}}$ using a linear regression procedure via solving

$$\underset{\boldsymbol{W}_{\text{FCN}}}{\text{argmin}} \quad \left\|\boldsymbol{Y}_0^{\text{train}} - \boldsymbol{X}_0^{(\text{ke})}\boldsymbol{W}_{\text{FCN}}^{(0)}\right\|_F^2 + \gamma\left\|\boldsymbol{W}_{\text{FCN}}^{(0)}\right\|_F^2 \tag{4}$$

where $\|\cdot\|_F$ indicates the Frobenius norm, and γ regularizes the objective function, with an optimal solution

$$\hat{\boldsymbol{W}}_{\text{FCN}}^{(0)} = (\boldsymbol{X}_0^{(\text{ke})\text{T}}\boldsymbol{X}_0^{(\text{ke})} + \gamma\boldsymbol{I})^{-1}\boldsymbol{X}_0^{(\text{ke})\text{T}}\boldsymbol{Y}_0^{\text{train}} \tag{5}$$

where $\hat{\boldsymbol{W}}_{\text{FCN}}^{(0)}$ indicates the estimated weight of the FCN layer, and $\cdot^{\text{T}}$ is the matrix transpose operator.

3.2. Few-shot Class Incremental Learning

Upon completing the AInit process, the FSCIL begins. Let

$$\boldsymbol{Y}_{0:k-1}^{\text{train}} = \begin{bmatrix} \boldsymbol{Y}_0^{\text{train}} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Y}_1^{\text{train}} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \vdots & & & & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{Y}_{k-1}^{\text{train}} \end{bmatrix}, \boldsymbol{X}_{0:k-1}^{\text{train}} = \begin{bmatrix} \boldsymbol{X}_0^{(\text{ke})} \\ \boldsymbol{X}_1^{(\text{ke})} \\ \vdots \\ \boldsymbol{X}_{k-1}^{(\text{ke})} \end{bmatrix}$$

where the structure of $\boldsymbol{Y}_{0:k-1}^{\text{train}}$ is sparse due to disjoint tasks, and $\boldsymbol{X}_j^{(\text{ke})}$ is the j^{th} embedding via

$$\boldsymbol{X}_j^{(\text{ke})} = g_{\{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_I\}}(f_{\text{CNN}}(\boldsymbol{X}_j^{\text{train}}, \boldsymbol{W}_{\text{CNN}})). \tag{6}$$

Without loss of generality, the learning problem in (4) given $\mathcal{D}_{0:k-1}^{\text{train}}$ can be expanded to

$$\underset{\boldsymbol{W}_{\text{FCN}}^{(k-1)}}{\text{argmin}} \quad \left\|\boldsymbol{Y}_{0:k-1}^{\text{train}} - \boldsymbol{X}_{0:k-1}^{\text{train}}\boldsymbol{W}_{\text{FCN}}^{(k-1)}\right\|_F^2 + \gamma\left\|\boldsymbol{W}_{\text{FCN}}^{(k-1)}\right\|_F^2 \tag{7}$$

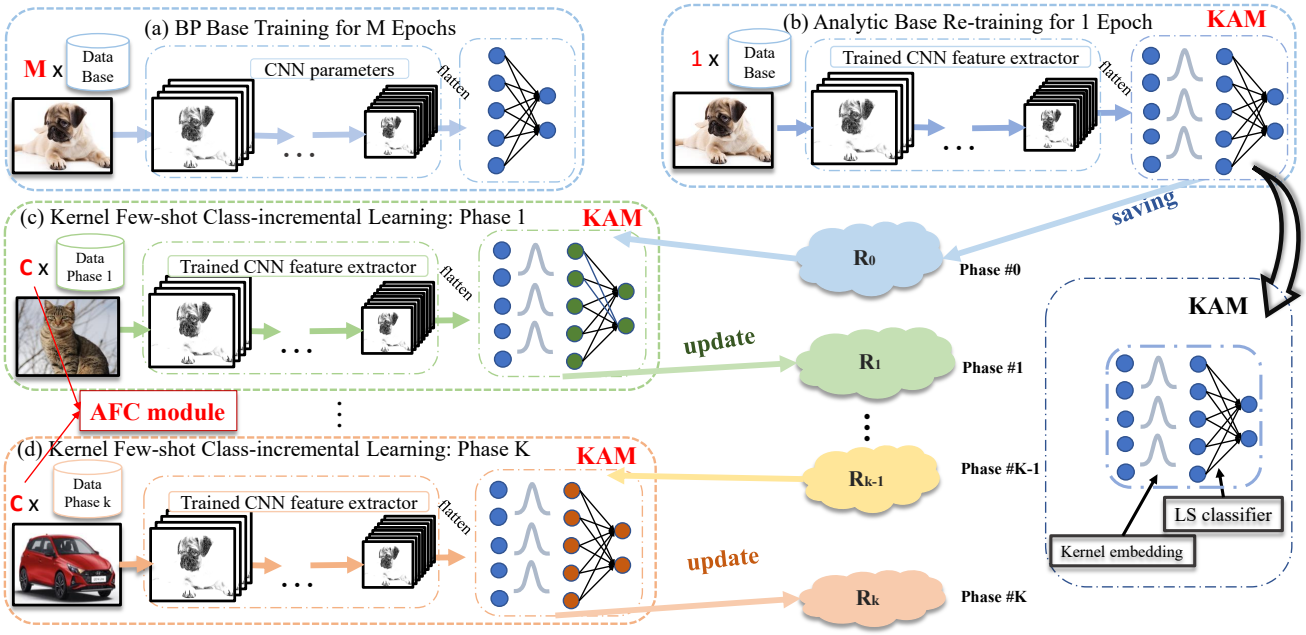

Figure 2. An overview of the proposed GKEAL. (a) The network is first trained on the base task for M epochs, which is then re-trained for 1 epoch adopting the KAM (a two-layer network including a kernel embedding process and an analytic LS classifier). (c,d) The FSCIL begins by adopting the pre-trained CNN backbone as feature extractor. The new tasks are learned in a recursive manner. In particular, the AFC module (with a parameter C) is adopted to balance base-new task preference.

which leads to

$$\hat{\boldsymbol{W}}_{\text{FCN}}^{(k-1)} = (\boldsymbol{X}_{0:k-1}^{(\text{ke})\text{T}} \boldsymbol{X}_{0:k-1}^{(\text{ke})} + \gamma \boldsymbol{I})^{-1} \boldsymbol{X}_{0:k-1}^{(\text{ke})\text{T}} \boldsymbol{Y}_{0:k-1}^{\text{train}}. \quad (8)$$

Our objective of FSCIL is to further learn new tasks on $\mathcal{D}_k^{\text{train}}$ given a network pre-trained on $\mathcal{D}_{0:k-1}^{\text{train}}$. Let $\boldsymbol{R}_k = (\boldsymbol{X}_{0:k}^{(\text{ke})\text{T}} \boldsymbol{X}_{0:k}^{(\text{ke})} + \gamma \boldsymbol{I})^{-1}$, the FSCIL can be solved via the following Theorem.

Theorem 3.1. Given $\mathcal{D}_k^{\text{train}}$, $\boldsymbol{R}_{k-1}$ and $\hat{\boldsymbol{W}}_{\text{FCN}}^{(k-1)}$ in (7), $\hat{\boldsymbol{W}}_{\text{FCN}}^{(k)}$ can be obtained recursively via

$$\hat{\boldsymbol{W}}_{\text{FCN}}^{(k)} = \Big[\underbrace{\hat{\boldsymbol{W}}_{\text{FCN}}^{k-1} - \boldsymbol{R}_k \boldsymbol{X}_k^{(\text{fe})\text{T}} \boldsymbol{X}_k^{(\text{fe})} \hat{\boldsymbol{W}}_{\text{FCN}}^{k-1}}_{\text{old tasks}} \quad \underbrace{\boldsymbol{R}_k \boldsymbol{X}_k^{(\text{ke})\text{T}} \boldsymbol{Y}_k^{\text{train}}}_{\text{new tasks}} \Big] \quad (9)$$

where

$$\boldsymbol{R}_k = \boldsymbol{R}_{k-1} - \boldsymbol{R}_{k-1} \boldsymbol{X}_k^{(\text{fe})\text{T}} (\boldsymbol{I} + \boldsymbol{X}_k^{(\text{fe})} \boldsymbol{R}_{k-1} \boldsymbol{X}_k^{(\text{fe})\text{T}})^{-1} \boldsymbol{X}_k^{(\text{fe})} \boldsymbol{R}_{k-1}. \quad (10)$$

Proof. See Supplementary material. □

Theorem 3.1 has indicated that the FSCIL through a recursive implementation based on $\mathcal{D}_k^{\text{train}}$ can reproduce the same weight obtained with a joint computation based on $\mathcal{D}_{0:k}^{\text{train}}$ in (7). That is, our proposed method as a FSCIL technique also possesses the same weight-invariant property as that in analytic learning [33].

As shown in (9), the solution expresses FSCIL in two folds: 1) The right part of weight is built for new tasks by taking only new information (i.e., $\boldsymbol{R}_k$, $\boldsymbol{X}_k^{(\text{ke})\text{T}}$ and $\boldsymbol{Y}_k^{\text{train}}$). 2) The left part absorbs both new (i.e., $\boldsymbol{R}_k$ and $\boldsymbol{X}_k^{(\text{ke})\text{T}}$) and base (i.e., $\hat{\boldsymbol{W}}_{\text{FCN}}^{(k-1)}$) knowledge. This pattern makes sense as the incremental learning should preserve the learned knowledge while accepting the new information's influence.

Why KAM and LS solution. The KAM and its LS solution are essential in terms of handling data-inefficiency scenarios where the FSCIL naturally belongs. The LS solution has anti-over-fitting nature [34], which is useful given very limited data during each incremental phase. The GKE process also tends to give good performance with small data availability [35].

Freezing the Backbone. As shown in (6), the kernelized embedding is obtained by feeding the input through the CNN backbone trained on the base dataset. That is, the backbone's parameters are not trainable during the FSCIL. Such a freezing action prevents the network from updating itself with new class features. This seems to result in certain performance decline, which is true in the traditional CIL case. However, in the few-shot scenario, data from new classes are scarce compared with that of the base training. These scarce samples are less likely to make vital contributions in improving the backbone's feature extraction power during the FSCIL. Such a decision (i.e., freezing the back-

bone) also happens in many other FSCIL methods (e.g., CEC [31]). It is a reasonable call to free the backbone in exchange for GKEAL's no-forgetting property for FSCIL.

3.3. Augmented Feature Concatenation For New Tasks

Theorem 3.1 has given an overview of the proposed GKEAL. However, unlike conventional incremental learning, FSCIL has the issue of sample imbalance. That is, the learning favors the base classes as the base samples are visited more frequently. Here we mitigate the imbalance via an augmented feature concatenation (AFC) process by amplifying the impact of new tasks.

In (9), instead of directly adopting $\boldsymbol{X}_k^{(\mathrm{fe})}$, the AFC process augments and concatenates the feature by

$$\bar{\boldsymbol{X}}_k^{(\mathrm{fe})} \Leftarrow \begin{bmatrix} g_{\{c_1,\ldots,c_I\}}(f_{\mathrm{CNN}}(\mathcal{A}_1(\boldsymbol{X}_k^{\mathrm{train}}), \boldsymbol{W}_{\mathrm{CNN}})) \\ g_{\{c_1,\ldots,c_I\}}(f_{\mathrm{CNN}}(\mathcal{A}_2(\boldsymbol{X}_k^{\mathrm{train}}), \boldsymbol{W}_{\mathrm{CNN}})) \\ \vdots \\ g_{\{c_1,\ldots,c_I\}}(f_{\mathrm{CNN}}(\mathcal{A}_C(\boldsymbol{X}_k^{\mathrm{train}}), \boldsymbol{W}_{\mathrm{CNN}})) \end{bmatrix}, \bar{\boldsymbol{Y}}_k^{\mathrm{train}} \Leftarrow \begin{bmatrix} \boldsymbol{Y}_k^{\mathrm{train}} \\ \boldsymbol{Y}_k^{\mathrm{train}} \\ \vdots \\ \boldsymbol{Y}_k^{\mathrm{train}} \end{bmatrix} \quad (11)$$

where we use $\Leftarrow$ to re-define $\boldsymbol{X}_k^{(\mathrm{fe})}$ and $\boldsymbol{Y}_k^{\mathrm{train}}$ for convenience. Here $\mathcal{A}_c(\boldsymbol{X}_k^{\mathrm{train}})$ indicates the c^{th} data augmentation on $\boldsymbol{X}_k^{\mathrm{train}}$. We use commonly seen augmentation techniques such as random horizontal flip, random cropping and normalizing. C here is referred to as the *augmentation count*. Note that the output $\bar{\boldsymbol{Y}}_k^{\mathrm{train}}$ is a concatenated matrix stacked C times with the original label. This resembles the multi-epoch training where data for each epoch are augmented randomly, but is not the same since the AFC process updates the weights "in one go" after concatenating the augmented features. Here $C \in \mathbb{Z}$ is an additional hyperparameter balancing the knowledge between old and new tasks. This rewrites (9) as

$$\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{(k)} = \begin{bmatrix} \hat{\boldsymbol{W}}_{\mathrm{FCN}}^{k-1} - \boldsymbol{R}_k \bar{\boldsymbol{X}}_k^{(\mathrm{fe})\mathrm{T}} \bar{\boldsymbol{X}}_k^{(\mathrm{fe})} \hat{\boldsymbol{W}}_{\mathrm{FCN}}^{k-1} & \boldsymbol{R}_k \bar{\boldsymbol{X}}_k^{(\mathrm{ke})\mathrm{T}} \bar{\boldsymbol{Y}}_k^{\mathrm{train}} \end{bmatrix}. \quad (12)$$

By augmenting the few-shot data of new tasks, the solution can avoid being too focused on base classes. For demonstration purpose, we have $\bar{\boldsymbol{X}}_k^{(\mathrm{fe})\mathrm{T}} \bar{\boldsymbol{X}}_k^{(\mathrm{fe})} \approx C \boldsymbol{X}_k^{(\mathrm{fe})\mathrm{T}} \boldsymbol{X}_k^{(\mathrm{fe})}$, $\bar{\boldsymbol{X}}_k^{(\mathrm{ke})\mathrm{T}} \bar{\boldsymbol{Y}}_k^{\mathrm{train}} \approx C \boldsymbol{X}_k^{(\mathrm{ke})\mathrm{T}} \boldsymbol{Y}_k^{\mathrm{train}}$ (augmented features are quite similar), which rewrites (12) as

$$\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{(k)} \approx \begin{bmatrix} \underbrace{\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{k-1} - C\boldsymbol{R}_k \boldsymbol{X}_k^{(\mathrm{fe})\mathrm{T}} \boldsymbol{X}_k^{(\mathrm{fe})} \hat{\boldsymbol{W}}_{\mathrm{FCN}}^{k-1}}_{\uparrow C \Rightarrow \downarrow \text{gain reduced}} & \underbrace{C\boldsymbol{R}_k \boldsymbol{X}_k^{(\mathrm{ke})\mathrm{T}} \boldsymbol{Y}_k^{\mathrm{train}}}_{\uparrow C \Rightarrow \uparrow \text{gain amplified}} \end{bmatrix}. \quad (13)$$

The update formula with AFC process points out that the output channels for new tasks (i.e., right-side weight in (13)) are amplified up to C times while the gains for old tasks are cut down (i.e., left-side weight in (13)). This analysis can later be supported in experiments (see Figure 4(d)).

The AFC works similarly to a regular data augmentation technique, but differs in that it augments the features and labels in matrix form and participates the calculation in a single shot. This allows certain interpretability such as that in (13). Regular data augmentation joins the calculation in a mini-batch form in a sequential manner, whose interpretability is buried during the iterations. We summarize the proposed GKEAL in Algorithm 1.

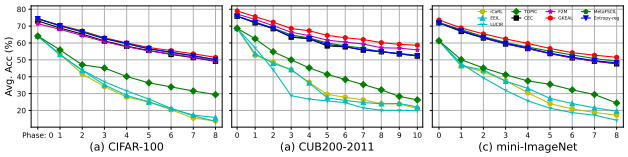

Figure 3. Average accuracy A_k w.r.t. each phase on (a) CIFAR-100, (b) CUB200-2011 and (c) *mini*-ImageNet. Details can be found in Table 1, Table 2 and Table 3.

Algorithm 1 GKEAL

Require: Data $\mathcal{D}_{0:K}^{\mathrm{train}}$, number of kernel vectors I, regularization parameter γ, width parameter β, augmentation count C.

1: **BP-based base training**: Train networks with BP on the base dataset.
2: **Analytic initialization (with $\mathcal{D}_0^{\mathrm{train}}$)**: i) Obtain kernelized embedding with (2) and (1); ii) Obtain base weight $\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{(0)}$ with (5). iii) Obtain and store $\boldsymbol{R}_0 = (\boldsymbol{X}_0^{(\mathrm{ke})\mathrm{T}} \boldsymbol{X}_0^{(\mathrm{ke})} + \gamma \boldsymbol{I})^{-1}$.
3: **for** $k = 1$ **to** K (with $\mathcal{D}_k^{\mathrm{train}}$, $\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{(k-1)}$ and $\boldsymbol{R}_{k-1}$) **do**
4: i) Obtain augmented kernelized embedding with (11);
5: ii) Update $\boldsymbol{R}_k$ with (10);
6: iii) Update weight $\hat{\boldsymbol{W}}_{\mathrm{FCN}}^{(k)}$ with (12);
7: **end for**

4. Experiments

In this section, we compare the proposed GKEAL with several state-of-the-art methods, including CIL-converted methods (i.e., iCaRL [20], EEIL [1], LUCIR [9]) and techniques specifically designed for FSCIL (i.e., TOPIC [23], CEC [31], F2M [22], MetaFSCIL [3] and Entropy-reg [14]). In addition, ablation study and parameter analysis are also included to reveal the contributions of GKEAL's components.

4.1. Experimental Setup

Dataset and Data Split. We evaluate the performance of GKEAL by training ResNet [8] on CIFAR-100 [12], CUB200-2011 [25] and *mini*-ImageNet [21], which have 100, 200 and 100 image classes respectively. These benchmark Data have image sizes of 32×32, 224×224 and 84×84 respectively. All compared methods follow the

Table 1. The accuracy among the compared methods on *mini*-ImageNet.

	Phase: 0	1	2	3	4	5	6	7	8	PD↓
iCaRL [20]	61.31	46.32	42.94	37.63	30.49	24.00	20.89	18.80	17.21	44.10
EEIL [1]	61.31	46.58	44.00	37.29	33.14	27.12	24.10	21.57	19.58	41.73
LUCIR [9]	61.31	47.80	39.31	31.91	25.68	21.35	18.67	17.24	14.17	47.14
TOPIC [23]	61.31	50.09	45.17	41.16	37.48	35.52	32.19	29.46	24.42	36.89
CEC [31]	72.00	66.83	62.97	59.43	56.70	53.73	51.19	49.24	47.63	24.37
F2M [22]	72.05	67.47	63.16	59.70	56.71	53.77	51.11	49.21	47.84	24.21
MetaFSCIL [3]	72.04	67.94	63.77	60.29	57.58	55.16	52.9	50.79	49.19	22.85
Entropy-reg [14]	71.84	67.12	63.21	59.77	57.01	53.95	51.55	49.52	48.21	23.63
GKEAL $(I, \beta, C = 10\text{k}, 10, 200)$	**73.59**	**68.90**	**65.33**	**62.29**	**59.39**	**56.70**	**54.20**	**52.59**	**51.31**	**22.28**

Table 2. The accuracy among the compared methods on CIFAR-100

	Phase: 0	1	2	3	4	5	6	7	8	PD↓
iCaRL [20]	64.10	53.28	41.69	34.13	27.93	25.06	20.41	15.48	13.73	50.37
EEIL [1]	64.10	53.11	43.71	35.15	28.96	24.98	21.01	17.26	15.85	48.25
LUCIR [9]	64.10	53.05	43.96	36.97	31.61	26.73	21.23	16.78	13.54	50.56
TOPIC [23]	64.10	55.88	47.07	45.16	40.11	36.38	33.96	31.55	29.37	34.73
CEC [31]	73.07	68.88	65.26	61.19	58.09	55.57	53.22	51.34	49.14	23.93
F2M [22]	71.45	68.10	64.43	60.80	57.76	55.26	53.53	51.57	49.35	**22.06**
MetaFSCIL [3]	**74.50**	70.10	66.84	62.77	59.48	56.52	54.36	52.56	49.97	24.53
Entropy-reg [14]	74.40	70.20	66.54	62.51	59.71	56.58	54.52	52.39	50.14	24.26
GKEAL $(I, \beta, C = 5\text{k}, 10, 200)$	74.01	**70.45**	**67.01**	**63.08**	**60.01**	**57.30**	**55.50**	**53.39**	**51.40**	22.61

split in [23]. The dataset is partitioned into a base training set (i.e., phase #0) and a sequence of incremental sets containing few-shot samples. For CIFAR-100 and *mini*-ImageNet, the base training set includes 60 classes. For CUB200-2011, the number of classes in the base training set is 100. The FSCIL is conducted in a 5-way 5-shot (5 classes with 5 samples in each class for each phase) manner for CIFAR-100/*mini*-ImageNet (total of 8 phases excluding the base training) and a 10-way 5-shot manner for CUB200-2011 (total of 10 phases excluding the base training).

Implementation details. For the training on the base dataset, we allow various training strategies for compared methods to achieve their desired performances. For instance, on CIFAR-100 F2M [22] requires 240 epochs for base training while the CEC [31], MetaFSCIL [3] and Entropy-reg [14] needs 100. For our GKEAL, for CIFAR-100 and *mini*-ImageNet, we train ResNet-20 and ResNet-18 on base datasets for 300 epochs with an initial learning rate of 0.1. The learning rate is then divided by 10 at 150, 225 and 275 epoch. For base training on CUB200-2011, following recent works [22, 31], we fine-tune a ResNet-18 pre-trained on ImageNet for 30 epochs with an initial learning rate of 0.01 which is divided by 10 at 15 epoch.

For hyperparameters, i.e., width parameter β, number of kernels I and augmentation count C, we run grid search on β={0.1,1,2,5,8,10,12,15,20,100}, I={1k,2k,5k,8k,10k,12k} (i.e., 1k indicates 1000) and C={1,50,100,150,200,250,300,350}. In particular, we adjust the search space of C for CUB200-2011 to C={1,2,5,8,10,12,15,20} because its base training contains much less data samples compared with those on CIFAR-100 or *mini*-ImageNet. As the few-shot scenarios are manually constructed, we use the remaining training data of the original few-shot classes as validation set. We report the result on the testing set using the best model on the validation set after conducting the search. For the regularization parameter γ, we fix it at $\gamma = 1$. All experiments are conducted using one RTX 2080Ti GPU with the results reported by the average of 3 runs. The results of compared methods are cited from [23], [31] and [22].

Evaluation Protocol. Following [23], the performance for the k^{th} phase is evaluated by the average accuracy, i.e., the test accuracy on the seen classes (i.e., $\mathcal{D}_{0:k}^{\text{test}}$), denoted by A_k. Also, the performance drop rate (PD), i.e., PD $= A_0 - A_K$, is included for evaluation. Reporting the performance drop is meaningful as some methods may give good results mainly due to a well-trained network on the base dataset. For results in ablation study and parameter analysis, we report the last-phase accuracy A_K for convenience.

4.2. Comparison with State-of-the-arts

As an overview, we depict the compared methods' accuracy evolution of CIFAR-100 (Figure 3(a)), CUB200-2011 (Figure 3(b)) and *mini*-ImageNet (Figure 3(c)) with respect to (w.r.t.) each phase. All methods experience decreasing accuracies. The reason behind this degradation is twofold. Firstly, the network takes in new data classes in each phase, giving it more choices to distinguish from, naturally leading to accuracy decrease. The second reason is FSCIL's

Table 3. The accuracy among the compared methods on CUB200-2011.

	Phase: 0	1	2	3	4	5	6	7	8	9	10	PD↓
iCaRL [20]	68.68	52.65	48.61	44.16	36.62	29.52	27.83	26.26	24.01	23.89	21.16	47.52
EEIL [1]	68.68	53.63	47.91	44.20	36.30	27.46	25.93	24.70	23.95	24.13	22.11	46.57
LUCIR [9]	68.68	57.12	44.21	28.78	26.71	25.66	24.62	21.52	20.12	20.06	19.87	48.81
TOPIC [23]	68.68	62.49	54.81	49.99	45.25	41.40	38.35	35.36	32.22	28.31	26.26	42.40
CEC [31]	75.85	71.94	68.50	63.50	62.43	58.27	57.73	55.81	54.83	53.52	52.28	23.57
F2M [22]	77.13	73.92	70.27	66.37	64.34	61.69	60.52	59.38	57.15	56.94	55.89	21.24
MetaFSCIL [3]	75.90	72.41	68.78	64.78	62.96	59.99	58.30	56.85	54.78	53.82	52.64	23.26
Entropy-reg [14]	75.90	72.14	68.64	63.76	62.58	59.11	57.82	55.89	54.92	53.58	52.39	23.51
GKEAL ($I, \beta, C = 10\text{k}, 15, 10$)	**78.88**	**75.62**	**72.32**	**68.62**	**67.23**	**64.26**	**62.98**	**61.89**	**60.20**	**59.21**	**58.67**	**20.21**

main concern, i.e., catastrophic forgetting, causing accuracy drop . As observed in the Figure 3, the CEC and F2M are giving similarly competitive results, outperforming iCaRL, EEIL, LUCIR and TOPIC by large margins. This is a reasonable observation as the iCaRL, EEIL and LUCIR are not specifically designed for FSCIL tasks, so they suffer rather significantly from over-fitting. For instance, the iCarL, a rather strong baseline in traditional CIL, achieves the average accuracy below 20%. The TOPIC is the first FSCIL baseline, giving a slightly better performance over that of the CIL methods, but cannot compete with the recent FSCIL techniques (e.g., CEC and F2M). For the most recent FSCIL teniques like MetaFSCIL and Entropy-reg, they slightly outperforms CEC and F2M.

Our GKEAL even outperforms the results of MetaFSCIL and Entropy-reg by a considerable amount (see gaps between the red curves and the second best curves in Figure 3), showing improved accuracy in each phase on all three datasets. The hyperparameters selected for CIFAR-100, CUB200-2011 and *mini*-ImageNet are $\{I, \beta, C = 5k, 10, 200\}$, $\{I, \beta, C = 10k, 15, 10\}$ and $\{I, \beta, C = 10k, 10, 200\}$ respectively. For convenience, the detailed results on *mini*-ImageNet are tabulated in Table 1 as a further support. The GKEAL achieves a 51.31% accuracy at the last phase, overtaking the second best result (MetaFSCIL's 49.19%) by 2.12%. In particular, the PD score of GKEAL is 22.28%, which is also the lowest, indicating a less forgetting among the compared methods. The detailed results for CIFAR-100 and CUB200-2011 are shown in Table 2 and Table 3, where the results show similar patterns.

4.3. Ablation Study and Parameter Analysis

Ablation Study. Here we conduct an ablation study to justify the contributions of the GKE module and AFC module. Both modules are important to GKEAL. In particular, the GKE gives a critical contribution. As shown in Table 4 (first 2 rows), the analytic learning does not work with the original extracted features by the BP algorithm. That is, lacking the GKE module results in catastrophic accuracy loss (e.g., a drop of last-phase accuracy from 51.21% to 7.22%). This is surprising as a mere linear layer tuned by BP can have reasonable achievement. This is because the matrix

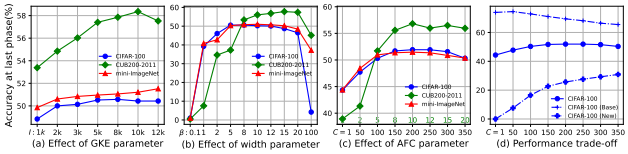

Figure 4. Performance with various (a) GKE parameter I, (b) width parameter β and (c) AFC parameter C. (d) The performance on base and new classes under various C values.

inverse experiences ill-conditioned scenario, leading to a breakdown of the solution. On the other hand, the AFC also contributes significantly. It improves the last-phase accuracy by 6.42% from 43.79% to 51.21%. However, the AFC alone cannot help improve the performance. This is because without the GKE, the analytic learning itself might go wrong (performance drop from 74.80% to 13.20%), AFC's new-old task balancing advantage is not demonstrated as a plug-in to our GKEAL. When GKE is applied, the AFC works effectively. As shown in Figure 4(d), the knowledge of base (old) classes and new classes are balanced by tuning the hyper-parameter C. Larger C emphasizes/weakens the base/new class knowledge, which well supports our theoretical claim in Eq. 13.

Hyperparameter Analysis. Upon proving the importance of the GKE and AFC modules, we further evaluate the impacts of the introduced kernel vector size I, width parameter β and augmentation count C. As shown in Figure 4(a), in general the GKEAL hungers a quite large I. The best performing parameters for CIFAR-100, CUB200-2011 and *mini*-ImageNet are at $I = 8k, 10k, 12k$. This is because the LS solution itself is prone to under-fitting, which can be compensated by an increased dimensionality. This requires an increased number of parameters, which is a common limitation of analytic learning. Investigating a more condense structure is in our future plans.

For parameter β in the GKE module, as shown Figure 4(b), there is a comfortable range for β at around $\beta \in [5, 15]$ for CIFAR-100/*mini-ImageNet* that gives good results, exceeding either bound would cause performance decay. In particular, too small or too large β values corrupt the training process (giving very low accuracy performance). For

Table 4. Ablation study of the GKE (w: $I = 5k$, w/o: removed) and AFC (w: $C = 200$, w/o: $C = 1$) modules.

GKE	AFC	Phase: 0	1	2	3	4	5	6	7	8
×	×	13.20	11.99	11.29	10.01	9.39	9.22	8.81	8.10	7.99
×	✓	12.56	10.80	10.29	9.81	9.36	8.60	8.00	7.89	7.22
✓	×	**74.80**	68.98	64.11	59.35	55.78	52.28	49.08	47.02	43.79
✓	✓	74.35	**70.32**	**66.21**	**62.37**	**60.01**	**56.98**	**55.12**	**53.39**	**51.21**

CUB200-2011, the performance peaks at around $\beta = 15$.

For parameter C, as shown in Figure 4(c), training on CIFAR-100/*mini*-ImageNet achieves the best results at around $C = 200$. Although training on CUB200-2011 prefers a much smaller $C = 10$, there is no intrinsic difference among these datasets. This is because CIFAR-100/*mini*-ImageNet has 500 samples per base class while CUB200-2011 has only 30. The augmentation ratios w.r.t. the base dataset are $\frac{200}{500} = 0.4$ for CIFAR-100/*mini*-ImageNet and $\frac{10}{30} = 0.33$ for CUB200-2011, which are close enough among these three datasets. In particular, we include the accuracies of CIFAR-100 reported on the base dataset $\mathcal{D}_0^{\text{test}}$ and the new class dataset $\mathcal{D}_{1:K}^{\text{test}}$ separately (see the dash lines in Figure 4(d)). We observe a consistent increasing (decreasing) performance pattern for base (new) classes with a larger C value, indicating that the AFC indeed balances the focus between the base and new classes.

5. Conclusion

In this paper, we propose Gaussian kernel embedded analytic learning (GKEAL) to handle the few-shot class incremental learning task. One key component of GKEAL is the kernel analytic module, containing a Gaussian embedding process which re-embeds the feature trained on the base dataset to produce more discriminative embeddings and a least-square classifier. The augmented feature concatenation module is another key contribution that balances base-new knowledge to enhance overall performance. Our experiments have conducted various empirical analysis (e.g., ablation study and parameter analysis), showing outstanding performance compared with the state-of-the-art methods.

Contributions

The contributions of authors in this work can be described as follows. Huiping Zhuang proposed the idea of GKEAL and did the experiments with Zhenyu Weng and Run He. Huiping Zhuang, Zhenyu Weng, Zhiping Lin and Ziqian Zeng contributed to the paper writing. Huiping Zhuang and Run He contributed to the rebuttal and revised the paper together with Zhiping Lin and Ziqian Zeng. All the authors contributed to the paper structure.

References

[1] Francisco M. Castro, Manuel J. Marin-Jimenez, Nicolas Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 1, 2, 5, 6, 7

[2] Ali Cheraghian, Shafin Rahman, Pengfei Fang, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Semantic-aware knowledge distillation for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2534–2543, June 2021. 2

[3] Zhixiang Chi, Li Gu, Huan Liu, Yang Wang, Yuanhao Yu, and Jin Tang. Metafscil: A meta-learning approach for few-shot class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14166–14175, June 2022. 5, 6, 7

[4] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 86–102. Springer, 2020. 2

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 06–11 Aug 2017. 2

[6] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[7] Ping Guo, Michael R Lyu, and NE Mastorakis. Pseudoinverse learning algorithm for feedforward neural networks. *Advances in Neural Networks and Applications*, pages 321–326, 2001. 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 5

[9] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 5, 6, 7

[10] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[11] 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

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[13] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. 1, 2

[14] Huan Liu, Li Gu, Zhixiang Chi, Yang Wang, Yuanhao Yu, Jun Chen, and Jin Tang. Few-shot class-incremental learning via entropy-regularized data-free replay. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, pages 146–162. Springer, 2022. 5, 6, 7

[15] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M. López, and Andrew D. Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2262–2268, 2018. 2

[16] Yaoyao Liu, Bernt Schiele, and Qianru Sun. An ensemble of epoch-wise empirical bayes for few-shot learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*. Springer International Publishing, 2020. 2

[17] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2544–2553, June 2021. 1, 2

[18] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[19] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991. 2

[20] 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 (CVPR)*, July 2017. 1, 2, 5, 6, 7

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 5

[22] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6747–6761. Curran Associates, Inc., 2021. 1, 2, 5, 6, 7

[23] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 5, 6, 7

[24] Kar-Ann Toh. Learning from the kernel and the range space. In *the Proceedings of the 17th 2018 IEEE Conference on Computer and Information Science*, pages 417–422. IEEE, June 2018. 2

[25] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[26] X. Wang, T. Zhang, and R. Wang. Noniterative deep learning: Incorporating restricted boltzmann machine into multilayer random weight neural networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(7):1299–1308, 2019. 2

[27] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[28] Han-Jia Ye, Hexiang Hu, and De-Chuan Zhan. Learning adaptive classifiers synthesis for generalized few-

shot learning. *International Journal of Computer Vision*, 129(6):1930–1953, 2021. 2

[29] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[30] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[31] Chi Zhang, Nan Song, Guosheng Lin, Yun Zheng, Pan Pan, and Yinghui Xu. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12455–12464, June 2021. 1, 2, 5, 6, 7

[32] Hanbin Zhao, Yongjian Fu, Mintong Kang, Qi Tian, Fei Wu, and Xi Li. Mgsvf: Multi-grained slow vs. fast framework for few-shot class-incremental learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. 2

[33] Huiping Zhuang, Zhiping Lin, and Kar-Ann Toh. Blockwise recursive Moore-Penrose inverse for network learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–14, 2021. 1, 2, 4

[34] Huiping Zhuang, Zhiping Lin, and Kar-Ann Toh. Correlation projection for analytic learning of a classification network. *Neural Processing Letters*, pages 1–22, 2021. 2, 4

[35] Huiping Zhuang, Zhiping Lin, and Kar-Ann Toh. Training multilayer neural networks analytically using kernel projection. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2021. 4

[36] Huiping Zhuang, Zhenyu Weng, Hongxin Wei, Renchunzi Xie, Kar-Ann Toh, and Zhiping Lin. ACIL: Analytic class-incremental learning with absolute memorization and privacy protection. In *Advances in Neural Information Processing Systems*, 2022. 3