Feature Estimations based Correlation Distillation for Incremental Image Retrieval

Wei Chen, Yu Liu, Nan Pu, Weiping Wang, Li Liu, *Senior Member, IEEE*, and Michael S. Lew

***Abstract*—Deep learning for fine-grained image retrieval in an incremental context is less investigated. In this paper, we explore this task to realize the model's continuous retrieval ability. That means, the model enables to perform well on new incoming data and reduce forgetting of the knowledge learned on preceding old tasks. For this purpose, we distill semantic correlations knowledge among the representations extracted from the new data only so as to regularize the parameters updates using the teacher-student framework. In particular, for the case of learning multiple tasks sequentially, aside from the correlations distilled from the penultimate model, we estimate the representations for all prior models and further their semantic correlations by using the representations extracted from the new data. To this end, the estimated correlations are used as an additional regularization and further prevent catastrophic forgetting over all previous tasks, and it is unnecessary to save the stream of models trained on these tasks. Extensive experiments demonstrate that the proposed method performs favorably for retaining performance on the already-trained old tasks and achieving good accuracy on the current task when new data are added at once or sequentially.**

***Index Terms*—Incremental learning, fine-grained image retrieval, correlations distillation, feature estimation**

I. Introduction

LEARNING is a life-long process for human beings so that we can learn continuously, devoid of forgetting previously acquired knowledge. However, this is not the case for deep neural networks, which suffer from the catastrophic forgetting problem, a phenomenon that occurs when a network is trained successively on a series of new tasks and the learning of these tasks degrades performance on previous tasks [1]. In particular, deep networks for image retrieval have been widely trained and validated on stationary datasets. As new data increase over time, these parameters pre-learned on the stationary datasets cannot be suited well for the non-stationary scenario. Nevertheless, human-like learning ability is required for deep networks based image retrieval.

The main challenge is to make the trained model adapt to new data without losing the knowledge on the seen data. Most conventional solutions for tackling this challenge suffer from obvious limitations. For example, one can use new data to fine-tune the model initialized from the optimal parameters pre-trained on old data. However, the model is driven towards the new data but forgets what was learned before. Alternatively, joint training achieves optimal retrieval performance on both old and new data, while it requires the presence of all the data during training. This requirement is hard to meet for several scenarios where legacy data are unrecorded due to privacy issues or simply too cumbersome to collect all the old data. Moreover, re-training old data may lead to an imbalance issue between the quantity of old data and that of new data [2][3].

Two incremental learning methods are developed to tackle the above limitations. First, the rehearsal based method utilizes generative adversarial nets (GANs) to synthesize samples *w.r.t.* previous data distributions [4][5]. This method faces the difficulty of generating images with complex semantics. Second, the regularization based methods can either focus on network parameters or output activations. Parameters-based regularization methods estimate the parameter importance of previous tasks, then penalizes the drastic updates of these parameters when the model is learning a new task, constraining the important parameters to stay close to their old values. Activation-based regularization methods, relying on the teacher-student framework, constrain the teacher model and the student model have similar outputs (*e.g.* features or probabilities). Due to its effectiveness and flexibility, the regularization methods have been widely explored for tasks such as image classification [2][3][6], but are less-explored for image retrieval. Recently, Parshotam *et al.* [7] regularize the representations via a normalized cross-entropy loss, training with metric learning for vehicle identification and retrieval. Chen *et al.* [8] propose regularizing both the representations and probabilities by combining a maximum mean discrepancy loss and a knowledge distillation loss [9] for fine-grained image retrieval (FGIR) [10]. As depicted in Figure 1(a), they only store and use the penultimate model to transfer previously

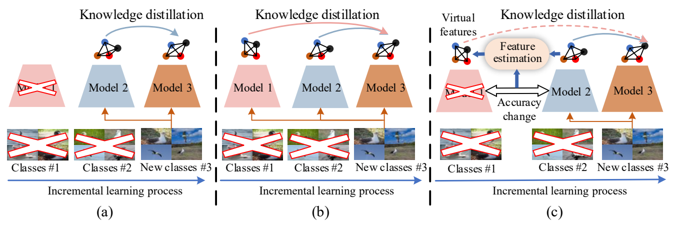

Fig. 1. Comparison of three knowledge distillation methods. We depict three steps of distillation. (a) Single-model distillation method only stores and uses the penultimate model; (b) Multi-model distillation method has to store all old models and distills from them more knowledge devoid of forgetting; (c) Our method only stores the penultimate model while can accumulate previous knowledge learned at each model through feature estimations.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the authors. The material includes more visualization of the retrieval results. This material is 2MB in size.

Wei Chen, Nan Pu and Michael S. Lew are with Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands.
Yu Liu is with DUT-RU International School of Information Science and Engineering, Dalian University of Technology, China.
Weiping Wang is with College of Systems Engineering, NUDT, China.
Li Liu is with College of Systems Engineering, NUDT, China, and with Center for Machine Vision and Signal Analysis, University of Oulu, Finland.
Corresponding author: Wei Chen, w.chen@liacs.leidenuniv.nl.

learned knowledge on old tasks, demonstrating experimentally that correlations distillation on the representations is more effective for reducing catastrophic forgetting on these tasks.

For the case where new tasks are added sequentially, which is referred to multi-task incremental learning, only distilling on the penultimate model is insufficient to reduce forgetting on all previous tasks [11]. In fact, transferring additional knowledge learned on previous tasks, *i.e.* , via multi-model distillation tackles this insufficiency, as shown in Figure 1(b). In multi-task incremental learning, a stream of deep models is produced as new tasks are added continuously. Each model is trained to learn the corresponding new task while remembers prior knowledge. However, it becomes too cumbersome and inefficient to store and use these models when more new data are added. Therefore, an arising question is that *how to use the model stream, not only the penultimate model, for knowledge distillation?* Few researchers address this problem in incremental tasks. Recently, a multi-model and multi-level knowledge distillation strategy is presented for incremental image classification [11]. However, the snapshots of all previous models still need to be saved and depend on network pruning methods to reconstruct. Thereby, the multi-level knowledge distillation is at the cost of a higher training complexity.

In this work, we face the above question to improve deep model's continuous retrieval ability. Semantic correlations of features are transferred as knowledge from a teacher model to a student model when new data are used only. For multi-task incremental learning, the model stream trained on preceding tasks is unnecessarily saved. Instead, we estimate representations for all previous models and further their semantic correlations, using the features extracted from the current new task (see Figure 1(c)). These correlations serve as an additional regularization to further prevent forgetting over previous tasks. Our contributions are summarized as follows:

(1) **Task contribution.** We focus on incremental image retrieval, a less-investigated task, by exploring semantic correlations of samples when training with new data only. Reducing catastrophic forgetting for image retrieval is challenging because small changes on retrieval features may have a big impact on the performance of feature matching.

(2) **Technical contribution**. We adopt feature correlations as knowledge for incremental learning. For multi-task incremental learning, we not only consider the knowledge from the penultimate model but also propose a simple yet effective feature estimation method to explore the feature correlations for the stream of models trained on previous tasks.

(3) **Empirical contribution**. Quantitative and qualitative experiments on two common benchmarks demonstrate that the proposed approach is effective for achieving optimal performance on both the old and new tasks when new incoming data are added at once or sequentially.

II. Related Work

Incremental image retrieval. Incremental learning can be categorized into architectural methods [3][6], rehearsal methods [4][5][12], and regularization methods [13][14][15][16]. It enables deep models to learn in a lifelong manner and has been studied for various tasks such as image classification [17][6][18], objection detection [19], image captioning [20], and semantic segmentation [21]. Most of them rely on classification probabilities to perform. In terms of reducing forgetting in an incremental setting, these tasks are forgiving and robust as long as features for old tasks are kept categorized into the range of classification decision boundaries. In contrast, retrieval tasks in an incremental setting are more challenging where the retrieval features are more important than classification probabilities because small changes on these features may have a big impact on the performance of feature matching and forgetting reducing. Recently, incremental retrieval have been explored in single-modal [8][22][23][24] and cross-modal [25][26][27], and we focus on single-modal image retrieval. IBL [22] and CIHR [23] are proposed to deal with the concept drift issue for hashing retrieval in non-stationary environments. However, the hash bits learned from hand-crafted features need to be pre-selected, and the trained hash functions for existing data need to be stored [22]. Also, selected images from previous training sessions are combined with new emerging images to train hash tables [23]. DIHN [24] is explored for deep incremental hashing retrieval in which old data are used as a query set to avoid forgetting. Fine-grained incremental image retrieval is studied with only using new data in each incremental session [8]. However, knowledge is only transferred from the penultimate model, causing the insufficiency to remember previous knowledge when performing multi-task incremental learning. In this work, we further distill additional knowledge from the model stream via a simple yet effective feature estimation method when only using new data in each incremental session.

Knowledge distillation. In general, a knowledge distillation system consists of three components: knowledge types, distillation strategies, and teacher-student structures [28]. Knowledge can be distilled from the output of either the final classifier or the intermediate layers. The distillation strategies characterize the differences between the teacher model and the student model, which can be measured by cross-entropy [9], L1 distance [29][30], L2 distance [31][32][33], Gramian matrix [34][35], Kullback-Leibler (KL) divergence [9][36], and maximum mean discrepancy (MMD). For more details about knowledge distillation, we refer readers to a recent survey [28]. Knowledge distillation provides an effective way to retain the learned knowledge devoid of forgetting in incremental tasks. For instance, the teacher model's output logits are used as "soft" labels to transfer knowledge to the student model for incremental image classification [37], but it focuses on transferring knowledge from the penultimate model to the current one (*i.e.* one-teacher framework). For this, a multi-teacher structure has been developed for better knowledge distillation [11][38][39]. For example, Zhou *et al.* [11] introduce using all previous models to transfer multi-level knowledge to train current new tasks. To avoid a great memory storage requirement, they prune previous models to get several "necessary" parameters during each training session.

Correlation learning. Correlation learning has been widely used for multi-modal tasks to explore the relevance between different layers or data samples [40][41][42][43][44]. It focuses on the relations between feature representations rather than the features themselves. These relations enable models to explore rich contextual information of images such as [43] where three-level of correlations are integrated for optimal feature learning. Correlation learning has been combined into knowledge distillation, based on the observation that semantically similar inputs tend to elicit similar patterns. For example, Peng *et al.* [44] use a symmetric adjacency matrix to encode a knowledge graph with category correlations and transfer them via a semantic-visual mapping network. Park *et al.* [15] propose a relational knowledge distillation method to transfer the relations between instances. Similarly, Peng *et al.* [45] introduce using the congruence between instances as knowledge for distillation, which is beneficial for guiding the student model to learn the correlations between instances. Similarity between activations of input pairs can also be extracted as knowledge to transfer into the student model [46]. The successful applications of correlation learning for knowledge distillation encourage its exploration for incremental learning tasks. In this work, we demonstrate experimentally that correlations among feature representations, characterized by Gramian matrix [34][35], are more effective for transferring knowledge to mitigate catastrophic forgetting.

III. Correlations Distillation for Incremental Image Retrieval

A. Problem formulation

Given a fine-grained dataset with n classes $\mathcal{D} = \{(\mathbf{X}^c, y^c)|c = 1, 2, \cdots, n\}$, each class c includes different amount of images $\mathbf{X}^c$ and they share the same ground-truth label y^c. The label is used to select a positive x_p and a negative x_n images for an anchor image x_a in each training iteration [47]. A deep network $f(\cdot, \boldsymbol{\theta})$ learns representations $\boldsymbol{F} = f(\boldsymbol{X}, \boldsymbol{\theta})$ under the constraint of the triplet loss using hard sampling strategy, whose goal is to push away the distance $D(x_a, x_n) = ||f(x_a;\boldsymbol{\theta}) - f(x_n;\boldsymbol{\theta})||_2^2$ between x_n and x_a by a margin $\delta > 0$ compared to $D(x_a, x_p)$. Namely,

$$||f(x_a;\boldsymbol{\theta}) - f(x_p;\boldsymbol{\theta})||_2^2 + \delta < ||f(x_a;\boldsymbol{\theta}) - f(x_n;\boldsymbol{\theta})||_2^2 \quad (1)$$

Before incremental training, the network is well trained on the n old classes, converging at old parameters $\boldsymbol{\theta}_o$, *i.e.*,

$$\boldsymbol{\theta}_o = \underset{\boldsymbol{\theta}}{argmin}\; L_{triplet}(f_0(\mathbf{X}^c; \boldsymbol{\theta})) \quad (2)$$

where $L_{triplet}(x_a, x_p, x_n) = [\delta + D(x_a, x_p) - D(x_a, x_n)]_+$, as defined in Eq. 1. To train network f_0 incrementally, new data from m classes $\{(\mathbf{X}^{c'}, \mathbf{Y}^{c'})\}$ where $c' \in (n+1, n+2, ..., n+m)$ are added $(\{\mathbf{X}^c\} \cap \{\mathbf{X}^{c'}\} = \emptyset)$ at once or sequentially, corresponding to one-task and multi-task cases, respectively.

As an example, the one-task case is depicted in Figure 2. At the start of training on m new classes, f_0 is copied into two copies. One is frozen as a teacher net, and another is used as a temporary initialization $f_1^{'}$ for further training ($\boldsymbol{\theta}_o = \boldsymbol{\theta}_n^{'}$, including the parameters in the Backbone and Embedding Net in Figure 2). We *only* use the m new classes to train to obtain f_1. Thus, the core issue of one-task incremental retrieval is to make the model f_1 with new parameters $\boldsymbol{\theta}_n$ maintain a stable performance on the n old classes and achieve competitive

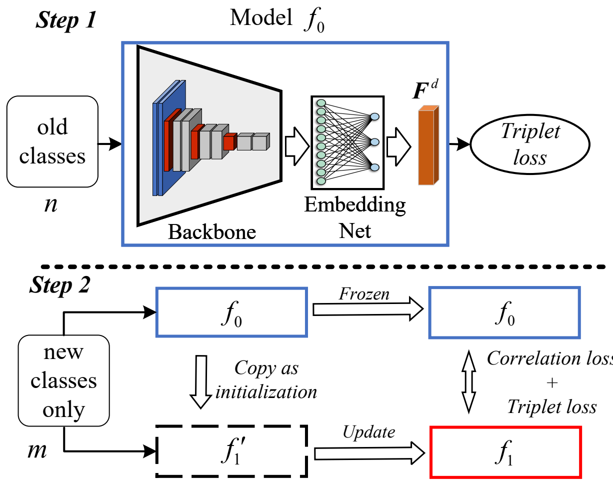

Fig. 2. One-task incremental learning includes two training steps. **Step 1**: a model f_0 is well trained in advance on the n old classes using ranking loss only. **Step 2**: the well-trained model f_0 is frozen as a teacher network. Meanwhile, the parameters of the Backbone and the Embedding Net included in this model f_0 are copied as initialization for a temporary model $f_1^{'}$, which is updated to the final model f_1 under the constraints of correlation loss and triplet loss. At Step 2, only the m new classes are used for training.

accuracy on the m new classes. Formally, the overall objective for this scenario is:

$$L(\mathbf{X}^{c'}; \boldsymbol{\theta}_o; \boldsymbol{\theta}_n) = \underbrace{\lambda_1 L_{triplet}(\mathbf{X}^{c'}; \boldsymbol{\theta}_n)}_{\text{for plasticity}} + \underbrace{\lambda_2 L_{corr}(\mathbf{X}^{c'}; \boldsymbol{\theta}_o; \boldsymbol{\theta}_n)}_{\text{for stability}} \quad (3)$$

where $L_{triplet}$ makes the model perform well on new tasks while L_{corr} is the correlation loss to stabilize prior performance. $\boldsymbol{\theta}_o$ and $\boldsymbol{\theta}_n$ are the parameters for old tasks and new tasks, respectively. λ_1 and λ_2 are the plasticity and stability hyper-parameters, which tune the influence of two loss terms.

B. Correlations distillation for one-task incremental learning

One-task incremental learning refers to the case that all the m new classes are added to the n old classes at once. As shown in Figure 2, the model $f_1^{'}$ serves as a to-be-trained student net. For the one-task incremental scenario, we propose to distill the semantic correlations as knowledge.

Specifically, the features with dimension d from the teacher model f_0 are formulated as $\boldsymbol{F}_o = f_0(\boldsymbol{X}^{c'}, \boldsymbol{\theta}_o) \in \mathbb{R}^{N\times d}$, and that from the student model f_1 are $\boldsymbol{F}_n = f_1(\boldsymbol{X}^{c'}, \boldsymbol{\theta}_n) \in \mathbb{R}^{N\times d}$, see Figure 2. Based on the fact that semantically similar inputs produce similar patterns in a trained network [46]. Therefore, a Gram matrix with a kernel function for $\boldsymbol{F}_o$ and $\boldsymbol{F}_n$ is defined:

$$G_o^{(i,j)} = \mathcal{K}(F_o^i, F_o^j);\; G_n^{(i,j)} = \mathcal{K}(F_n^i, F_n^j) \quad (4)$$

Here, we further define the function $\mathcal{K}(\cdot)$ as inner product, *i.e.*, $\mathcal{K}(F^i, F^j) = <F^i, F^j>$. Each entry (i, j) in $\boldsymbol{G} \in \mathbb{R}^{N\times N}$ represents the correlations of the same activation $(i = j)$ or these between different activations $(i \neq j)$. To compare the difference between $\boldsymbol{G}_o$ and $\boldsymbol{G}_n$, we first normalize these correlation matrices with Softmax function $\sigma(\cdot)$, and then use Kullback–Leibler divergence to characterize their difference, which is formulated as correlation loss L_{corr}.

$$L_{corr} = \frac{1}{N}\sum KL(\sigma(\boldsymbol{G}_o), \sigma(\boldsymbol{G}_n)) \quad (5)$$

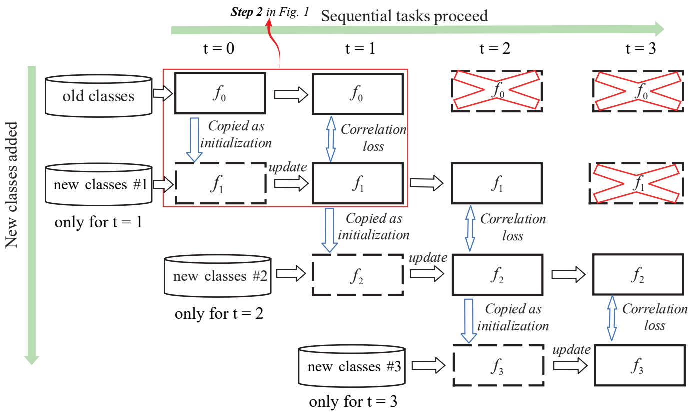

Fig. 3. Illustration of multi-task incremental learning when three groups of new classes are added sequentially (from task t=1 to task t=3). For each round when new classes are added, the model trained on a previous task is frozen its parameters as a teacher net and is also copied as initializations of the model for new classes. Each round can be viewed as one-task incremental learning. As the training proceeds, the previous models are not saved to simplify the training process. For simplicity, the triplet loss is ignored.

C. Feature estimation for multi-task incremental learning

Compared to the above one-task setting, the multi-task scenario is more complex where all m new classes are divided into t groups, $\boldsymbol{X}_0^{c'}$,..., $\boldsymbol{X}_t^{c'}$. For clarity, we illustrate its learning process in Figure 3. As more new classes added sequentially, the model, correspondingly, evolutes from the initial model f_0 to the current one f_t. In practice, it may be difficult to save the stream of models. For this limit, we only save the model trained on the penultimate task $t-1$ when proceeding current task t for t^{th} new classes $\boldsymbol{X}_t^{c'}$. For example, when training on the 3^{rd} group of new classes (task t=3), the knowledge is distilled only from the penultimate models f_2, while the previous models f_0 and f_1 are not saved. Due to the lack of previous models, it causes two drawbacks: (1) the knowledge is distilled only from the penultimate model f_{t-1} to the model on the current task t, and (2) the trained model f_t may forget more on old tasks prior to $t-1$. Therefore, it is natural to raise a question that how to utilize these unsaved models trained prior to the penultimate task $t-1$ for transferring additional knowledge to supervise the training of current task t.

Hereafter, for better understanding, we introduce the multi-task scenario by defining an adaptive model f_t for the current task t, a frozen model f_{t-1} trained on penultimate task $t-1$, and unsaved models f_{t-2}, ..., f_0 for earlier tasks $t-2$, ..., 0, as shown in Figure 4. Since the frozen model f_{t-1} is initialized from the previous unsaved model f_{t-2} at the start of training on task $t-1$, the feature distributions of these two models have some inherent relations, which can be reflected through their accuracy (*e.g.*, mAP). This accuracy evolution along with training the models stream gives a hint for feature estimation.

1) Accuracy drops and accuracy gains: We propose a simple yet effective method to estimate the feature distributions for all unsaved models, which serve as an additional regularization term for training on current task t ($t \geq 2$). For this purpose, we first focus on the accuracy change during training from task $t-2$ to task $t-1$. Parameters of the penultimate model f_{t-1} are copied from those of the model f_{t-2}. Before training on task $t-1$, the accuracy on its old tasks and the new classes $\boldsymbol{X}_{t-1}^{c'}$ are recorded as Acc_o^b and Acc_n^b, respectively. Naturally, Acc_n^b is far from accurate since the penultimate model f_{t-1} is not trained specifically for new data. After training on task $t-1$, the accuracy on these old tasks and new classes $\boldsymbol{X}_{t-1}^{c'}$ are recorded as Acc_o^a and Acc_n^a, respectively. Intuitively, the model f_{t-1} acquires new knowledge on new classes $\boldsymbol{X}_{t-1}^{c'}$, and the accuracy increases from Acc_n^b to Acc_n^a (*i.e.*, accuracy gains). In contrast, model f_{t-1} may degrade accuracy from Acc_o^b to Acc_o^a (*i.e.*, accuracy drops) because this model is driven towards the new data.

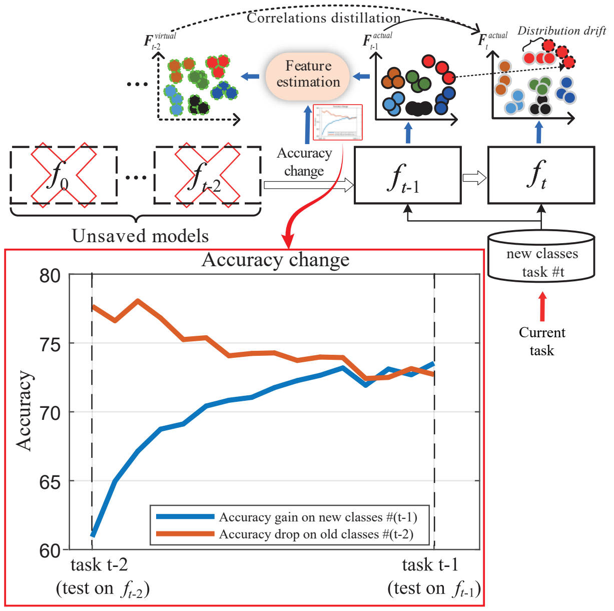

Fig. 4. Illustration of feature estimation when performing the task t. The virtual feature distribution of unsaved model f_{t-2} can be estimated by that of frozen model f_{t-1} under multi-task incremental learning.

The accuracy drops and accuracy gains, related to the stability-plasticity trade-off, are criteria that correspond to old tasks and new tasks, respectively. For instance, if a model has larger stability on previous tasks, both the accuracy drops and accuracy gains are small. In contrast, if the stability is too weak, the model suffers obvious accuracy drops and forgetting on previous tasks. Inspired by [48], we define the accuracy changes using the accuracy drops and accuracy gains:

$$\alpha_{drop} = \frac{(Acc_o^a - Acc_o^b)}{Acc_o^b}, \alpha_{gain} = \frac{(Acc_n^a - Acc_n^b)}{Acc_n^b} \quad (6)$$

As the training proceeds from task $t-2$ to task $t-1$, their accuracy changes continuously on old classes (the brown-color line in Figure 4) and new classes (the blue-color line). Rather than saving these models, we only need to record their accuracy drops $\alpha_{drop}|_{(t-2)\to(t-1)}$ and accuracy gains $\alpha_{gain}|_{(t-2)\to(t-1)}$, which are meta-data of these models and provide implicit information to estimate the feature distribution drifts. Here, the subscript "$(t-2) \to (t-1)$" means the knowledge is distilled from task $t-2$ to penultimate task $t-1$.

2) Distribution drifts estimation: Estimating feature distribution drifts was explored in [49] where the attribute vectors are learned based on the source set and target set, then the learned vectors are used to estimate new features. In this work,

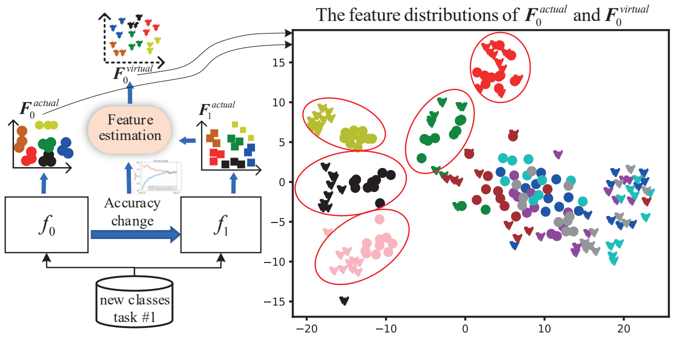

Fig. 5. We save model f_0 for validating its features qualitatively. The circle denotes the actual features from f_0. The triangle indicates the virtual features estimated by model f_1 according to the accuracy change from f_0 to f_1. The feature estimation block has been discussed in Eq. 7 and Eq. 8. We visualize 10 samples per class of a total 10 classes on the CUB-Birds dataset.

we estimate feature drifts via the change of model accuracy. We only save the penultimate model f_{t-1} when training on current task t, see Figure 4. The recorded accuracy change from model f_{t-2} to model f_{t-1} has been reflected through the drifts of their feature distributions. Based on this, we use the accuracy change (α_{drop}, α_{gain}) and the available features from the model f_{t-1} to estimate the feature drifts which are used to further compute virtual features for model f_{t-2}. To be specific, when feeding t^{th} group of new classes $\boldsymbol{X}_t^{c'}$ into the model f_{t-1} and the adaptive model f_t, we obtain their corresponding actual features $\boldsymbol{F}_{t-1}^{actual} = f_{t-1}(\boldsymbol{X}_t^{c'})$ and $\boldsymbol{F}_t^{actual} = f_t(\boldsymbol{X}_t^{c'})$. Since the accuracy drops and accuracy gains from model f_{t-2} to model f_{t-1} have been obtained, we estimate their feature distribution drifts using a simple yet effective method:

$$\begin{aligned}&\boldsymbol{\Delta}|_{(t-2)\to(t-1)} \approx \boldsymbol{\alpha}\cdot \boldsymbol{F}_{t-1}^{actual}\\ &s.t.\ \boldsymbol{\alpha} = Cat(\alpha_1,...,\alpha_i,...,\alpha_N),\ \alpha_i\in\mathbb{R}^d, \boldsymbol{\alpha}\in\mathbb{R}^{N\times d}\\ &\alpha_i \sim U(\alpha_{drop}|_{(t-2)\to(t-1)}, \alpha_{gain}|_{(t-2)\to(t-1)})\end{aligned} \quad (7)$$

where $Cat(\cdot)$ means vector concatenation operation. Each raw vector α_i is randomly sampled from the uniform distribution $U(\cdot,\cdot)$ according to α_{drop} and α_{gain}. Thereby, $\boldsymbol{\alpha}$ has the same dimension with the features $\boldsymbol{F}$. In theory, the expectation of each sampling in $\boldsymbol{\alpha}$ is close to $0.5\times(\alpha_{drop}+\alpha_{gain})$.

It is assumed that the features change uniformly during sequential training and the changes can be reflected through the defined accuracy drops and accuracy gains. With this hypothesis, the feature drifts $\boldsymbol{\Delta}|_{(t-2)\to(t-1)}$ can be evaluated according to actual features $\boldsymbol{F}_{t-1}^{actual}$. With the feature drifts, inspired by [49], the virtual feature distributions for unsaved model f_{t-2} are estimated:

$$\boldsymbol{F}_{t-2}^{virtual} = \boldsymbol{F}_{t-1}^{actual} + k\,\boldsymbol{\Delta}|_{(t-2)\to(t-1)} \quad (8)$$

where k is a scaling factor, we set $k=1$. The reason why we can estimate the virtual features $\boldsymbol{F}_{t-2}^{virtual}$ from $\boldsymbol{F}_{t-1}^{actual}$ is because the parameters of model f_{t-1} are initialized from model f_{t-2} at the start of training f_{t-1}.

Similarly, we can further approximate the virtual feature $\boldsymbol{F}_{t-3}^{virtual}$ for model f_{t-3} according to the already-estimated $\boldsymbol{F}_{t-2}^{virtual}$, its accuracy drops $\alpha_{drop}|_{(t-3)\to(t-2)}$ and accuracy gains $\alpha_{gain}|_{(t-3)\to(t-2)}$ from task $t-3$ to task $t-2$. Normally,

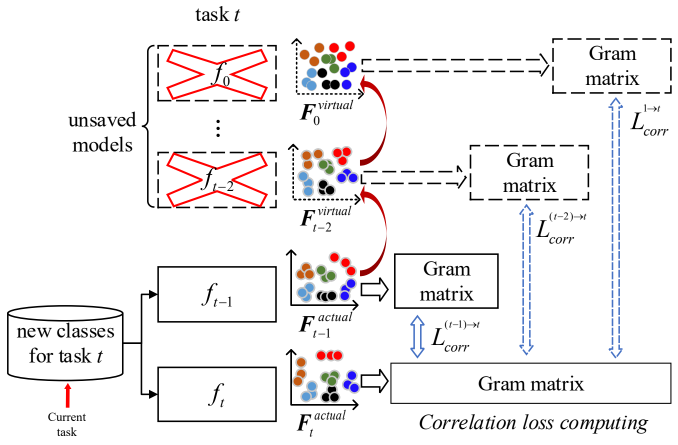

Fig. 6. Feature estimations based knowledge distillation. The red arrows denote feature estimation process. The dash arrows indicate the features are virtually estimated from the actual features. For instance, the superscript "$(t-2)\to t$" refers to the Gram matrix of task $(t-2)$ is used as supervision for training current task t, which is formally defined in Eq. 5.

with a recursive scheme, the virtual features of all previous unsaved models can be estimated using their recorded accuracy drops, accuracy gains, and already-estimated virtual features. Finally, the features of first model f_0 are estimated as:

$$\begin{aligned}\boldsymbol{F}_0^{virtual} &= (1+k\,\boldsymbol{\alpha}|_{(t-2)\to(t-1)})(1+k\,\boldsymbol{\alpha}|_{(t-3)\to(t-2)})\\ &\quad ...(1+k\,\boldsymbol{\alpha}|_{(0)\to(1)})\boldsymbol{F}_{t-1}^{actual}\end{aligned} \quad (9)$$

Qualitative Validation. We test the hypothesis in Eqs. 7,8 by visualizing and comparing the feature distributions when training task $t=2$ for the 2^{nd} group of new classes in Figure 5. To do this, we save model f_0 to visualize its output features. The output features from model f_1 are fed into the feature estimation module (defined in Eq. 7 and Eq. 8) according to the range of accuracy changes when training from model f_0 to model f_1. Their feature distributions are illustrated in Figure 5. For almost half of these classes, the estimated virtual features distribute closely to the actual features from target model f_0, demonstrating that the estimated features can also provide supervisory information to some extent.

3) Importance for estimated features: The estimated features for all previous unsaved models serve as additional regularization terms. Thus, more Gram matrices $\boldsymbol{G}^{virtual}$ are computed based on these estimated features, as illustrated in Figure 6. To this end, the additional correlation loss, such as $L_{corr}^{(t-2)\to t}$, based on the estimated features is formulated as:

$$L_{corr}^{(t-2)\to t} = \frac{1}{N}\sum\left(KL(\sigma(\boldsymbol{G}_{t-2}^{virtual}), \sigma(\boldsymbol{G}_t^{actual}))\right) \quad (10)$$

When more new classes are added sequentially, more Gram matrices are computed through the recursively-estimated features. However, these Gram matrices cannot be treated identically when used to distill knowledge for training current task t since the accumulated errors may make the recursively estimated features more and more unreliable. For tackling this limitation, the estimations for earlier tasks are assigned with smaller importance. Naturally, the importance is related to the indices of old tasks. To this end, we formulate the correlation loss terms with different importance factors:

$$L_{corr} = L_{corr}^{(t-1)\to t} + \underbrace{\frac{1}{(t-1)} L_{corr}^{(t-2)\to t} + \frac{0.1}{(t-2)} L_{corr}^{(t-3)\to t} + ... + \frac{(0.1)^{t-2}}{1} L_{corr}^{1\to t}}_{\text{Feature estimation for prior sequential tasks(t}\geq 2)} \tag{11}$$

For one-task incremental scenario (t=1), Eq. 11 can be re-written as Eq. 5. If more tasks are performed ($t \geq 2$), each semantic correlation loss based on the estimated virtual features are constrained with importance factors ($\frac{1}{(t-1)}$, $\frac{0.1}{(t-2)}$,...). Substituting the term Eq. 11 into Eq. 3, we obtain the overall objective function for incremental FGIR.

IV. Experiments

A. Datasets and experimental setup

We evaluate the proposed method on two fine-grained datasets: CUB-Birds-200 [50] and Stanford-Dogs-120 [51]. To build training sets and testing sets, we choose 60% images from each sub-category as training sets and 40% as testing sets. Afterwards, we split the first 100 sub-categories (60 for the Dogs dataset) as the old classes (*i.e.*, n=100 or 60) and the remaining 100 (60 for the Dogs dataset) sub-categories as new classes (*i.e.*, m=100 or 60), which are added at once or sequentially. For the sequential case, these new classes are divided into several groups evenly. Note that all splits are in the order of official classes. In the following text, we use the class index of each dataset to denote a group of new classes. For example, "*classes (101-125)*" in italic means that the m=25 new classes from the index 101 to 125 are used for training, the corresponding trained model is f_1*(101-125)*. For clarity, the details of datasets are reported in Table I.

TABLE I
Statistics of two datasets used in our experiments.

Datasets	Training set (#Image/#Class)			Testing set (#Image/#Class)		
	Old cls.	New cls.	Total	Old cls.	New cls.	Total
CUB-200	3504/100	3544/100	7048/200	2360/100	2380/100	4740/200
Stanford-120	6000/60	6000/60	12000/120	4561/60	3929/60	8580/120

Implementation details. We utilize Google Inception as a backbone net. The whole process includes two stages: initial model training and incremental training. In the first stage, the initial model f_0 is trained to converge on the n old classes using triplet loss only and optimized by the Adam optimizer with a learning rate of 1×10^{-6} and a batch size of 80, while the embedding net (*i.e.*, fully-connected layers) is updated with a learning rate of 1×10^{-5}. In the second stage, we use the converged model f_0 as initialization and train a new model f_1 on the images from m new classes using Eq. 3, with the same learning rate in the first stage. The initial model f_0 trained on the n old *classes (1-100)* or *(1-60)* is re-wrote as f_0*(1-100)* or f_0*(1-60)*. Likewise, the model f_1 is represented by the added m new classes, such as f_1*(101-200)* or f_1*(61-120)* for one-task incremental scenario. We follow the sampling strategy in [52] and each incremental process is trained 800 epochs until convergence. Following the practice in [53][52], the output 512-d features (F^d in Figure 2) are used for retrieval. Moreover, the margin in triplet loss is $\delta = 0.5$. Our code is available at https://github.com/cw1091293482/Deep-Incremental-Image-Retrieval.

Evaluation metrics. We use the Recall@1 [53][54] and mean Average Precision (mAP) as retrieval metrics, and use average incremental accuracy [6][55] and average forgetting [48] to evaluate incremental learning.

B. Hyper-parameter selection

We first analyze the hyper-parameter λ_1 and λ_2 in Eq. 3. They are critical for achieving a trade-off performance on n old classes and m new classes. Since we explore the stability of the deep model, we fix $\lambda_1 = 1$ and discuss the impact of λ_2 when all new *classes (101-200)* are added at once (*i.e.*, m = 100). The results of trained model f_1*(101-200)* on the CUB-Birds dataset are reported in Table II. "Initial model" refers to the model f_0 that is trained on the n old *classes (1-100)* and is directly tested on the m new *classes (101-200)*, whereas "Reference" denotes the model jointly trained on the $(n+m)$ classes. According to the averaged accuracy, the best performing factor is $\lambda_2 = 10$ where a trade-off performance on the respective old classes and new classes is achieved. Therefore, for the following comparison, we choose to set hyper-parameter $\lambda_1 = 1$ and $\lambda_2 = 10$ for our method.

TABLE II
Sensitivity Analysis of parameter λ_2 on the CUB-Birds dataset. Note that λ_1 is fixed to 1.

Configuration	Old *(1-100)*		New *(101-200)*		Average	
	Recall@1	mAP	Recall@1	mAP	Recall@1	mAP
Initial model	79.24	55.78	46.93	19.54	63.09	37.66
$\lambda_2 = 25$	79.24	55.06	68.91	37.84	74.08	46.45
$\lambda_2 = 20$	78.81	54.78	71.30	40.35	75.06	47.57
$\lambda_2 = 15$	77.97	53.81	72.56	42.86	75.27	48.34
$\lambda_2 = 10$	**77.71**	**52.25**	**75.00**	**46.51**	**76.36**	**49.38**
$\lambda_2 = 5$	75.68	48.43	76.47	48.63	76.08	48.53
$\lambda_2 = 1$	72.16	43.28	76.05	49.22	74.11	46.25
$\lambda_2 = 0.1$	72.03	42.60	75.84	49.24	73.94	45.92
Reference	78.18	52.17	79.24	50.99	78.71	51.58

C. One-task scenario evaluation

Baselines. IBL [22], CIHR [23], and DIHN [24] have been explored for incremental hashing retrieval. The main difference with ours is that they used old data for training to avoid forgetting, while we use new data *only*. We take [8] as a baseline, which took feature-level and probability-level regularization for FGIR. For a fair comparison, we consider feature-level regularization (*i.e.*, maximum mean discrepancy loss). We also compare to the popular algorithms including EWC[1], ALASSO[2], NCE loss[3], and L2 loss. Specifically, EWC [17] and ALASSO [16] are the network parameters regularization methods. To deploy these methods, we further train a classifier on the top of the embedding net so that these two methods can be reproduced correctly. NCE loss [7] regularizes the inner product of an anchor-positive feature pair

[1]https://github.com/joansj/hat/tree/master/src/approaches
[2]https://github.com/dmpark04/alasso
[3]https://github.com/ProsusAI/continual-object-instances

TABLE III
RECALL@1 AND MAP (%) OF INCREMENTAL FGIR TRAINED FOR THE ONE-TASK SCENARIO, "INITIAL MODEL f_0" INDICATES MODEL TRAINED ON THE FIRST 100 CLASSES ON THE DATASETS (*i.e.*,"INITIAL MODEL f_0*(1-100)*). SIMILARLY, "REFERENCE MODEL" INDICATES THE MODEL f_0 TRAINED ON ALL CLASSES OF DATASETS (*i.e.*, "INITIAL MODEL f_0*(1-200)* OR f_0*(1-120)*"). THE BEST PERFORMANCE IS REPORTED IN BOLD.

Dataset	**CUB-Birds-200**						**Stanford-Dogs-120**					
Configuration and Results (%)	Old *classes (1-100)*		New *classes (101-200)*		Average		Old *classes (1-60)*		New *classes (61-120)*		Average	
	Recall@1	mAP	Recall@1	mAP	Recall@1	mAP	Recall@1	mAP	Recall@1	mAP	Recall@1	mAP
Initial model f_0*(1-100)* or f_0*(1-60)*	79.24	55.78	46.93	19.54	63.09	37.66	81.27	66.05	69.28	34.13	75.28	50.09
⇒ Model f_1 w fine-tuning	70.21	42.57	**75.13**	**48.90**	72.67	45.74	73.96	45.24	**83.69**	**67.25**	78.83	56.25
⇒ Model f_1 w EWC [17]	73.32	45.73	72.84	44.14	73.08	44.94	74.76	46.92	81.45	62.69	78.11	54.81
⇒ Model f_1 w ALASSO [16]	72.88	43.87	72.94	45.50	72.91	44.69	75.92	48.35	81.50	63.40	78.71	55.88
⇒ Model f_1 w NCE$_{EWC}$ [7]	72.63	43.80	73.07	45.15	72.85	44.48	75.12	47.88	81.62	62.99	78.37	55.44
⇒ Model f_1 w L2 loss [21]	75.93	50.23	74.12	47.47	75.03	48.85	78.99	56.57	83.23	66.63	81.11	61.60
⇒ Model f_1 w MMD loss [8]	77.03	51.10	74.12	45.05	75.58	48.08	79.49	**59.43**	83.35	65.21	81.42	**62.32**
⇒ Model f_1 w Our method	**77.71**	**52.25**	75.00	46.51	**76.36**	**49.38**	**79.92**	58.37	83.48	66.01	**81.70**	62.19
Reference model (joint training)	78.18	52.17	79.24	50.99	78.71	51.58	80.37	62.48	83.10	66.78	81.74	64.63

and 9 anchor-negative feature pairs via a normalized cross-entropy loss. This method is combined into EWC algorithm for incremental learning. We follow this protocol by mining 9 hard negative samples (termed as NCE$_{EWC}$). L2 loss [19][13][14][21] focuses on minimizing the Euclidean distance between the features from the teacher-student models. For a fair comparison, the above four methods are trained with triplet loss $L_{triplet}$, having the same hyper-parameter $\lambda_1 = 1$. In terms of the plasticity factor λ_2, we tune this factor for four methods in incremental FGIR until we get their **optimal** performance. As a result, the corresponding plasticity factors are tuned as 8000, 0.2, 10, and 0.1, respectively. Moreover, the "Reference" by joint learning serves as an *upper-bound* performance for all methods. The fine-tuning method is also used as a reference for the new tasks since there is no knowledge distillation regularization.

One-task incremental learning refers to the case that m new classes are added at once (m=100 or m=60). This case is similar to transfer learning, while incremental training further emphasizes reducing forgetting on the n old classes. The results are reported in Table III. Note that only model f_0 is available, thereby it is unnecessary to estimate virtual features.

Naturally, the initial model f_0 trained on the n old classes performs poorly on the m new unseen classes. Take the CUB-Birds dataset as an example, mAP is 19.54% when the initial model f_0*(1-100)* is tested on the m new classes without any re-training. Using the initial model f_0, we further re-train on the m new classes using different incremental algorithms to obtain the model f_1, whose performance is distinct on the old and new classes, as shown in Table III. The fine-tuning method achieves the best accuracy on the new classes, it improves the accuracy (19.54%→48.90% in mAP) on the new classes on the CUB-Birds dataset but degrades accuracy (*i.e.*, forgetting) on the old classes (55.78%→42.57% in mAP). Similar trends can be observed on the Stanford-Dogs dataset.

For other algorithms, the models trained by network parameters regularization methods such as EWC and ALASSO show a similar trend that they reduce forgetting on the n old classes, but their performance on the m new classes is less competitive compared to the fine-tuning method. NCE$_{EWC}$ regularises metric learning via cross-entropy loss on the feature embeddings. We find this method has some limited benefits. For example, it improves on the Stanford-Dogs dataset in

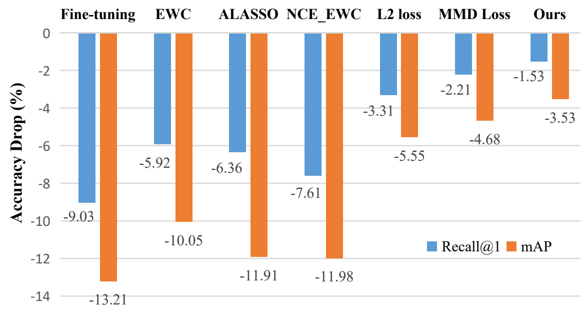

Fig. 7. Accuracy drop on the CUB-Birds dataset. The model f_1*(101-200)* trained by different incremental methods is tested on the same testing set.

terms of the average performance. L2 loss and MMD loss regularize the features directly. For L2 loss, it regularizes the model f_1 to forget less on the old classes of two datasets. For instance, on the CUB-Birds dataset, it reduces the degradation by 3.31% of Recall@1 (79.24%→75.93%) and 5.55% of mAP (55.78%→50.23%), see Table III for details.

MMD loss is more similar to our method in which feature correlations are also considered [8]. Compared to MMD loss, our method, in most cases, suffers less accuracy degradation on two datasets. For instance, our method degrades the Recall@1 on the n old classes by 1.53% (79.24%→77.71%) and 1.35% (81.27%→79.92%) on CUB-Birds and Stanford-Dogs, respectively, whereas the MMD loss degrades the Recall@1 on the old classes by 2.21% (79.24%→77.03%) and 1.78% (81.27%→79.49%) on two datasets. Moreover, in terms of the performance on the m new classes, our method also achieves closer accuracy to that of the fine-tuning method.

We visualize the accuracy drop of different methods on the n old classes on the CUB-Birds dataset for clarity. The results are shown in Figure 7, demonstrating that feature correlation matters for reducing accuracy degradation. Meanwhile, the accuracy gain for the n new classes is shown in Figure 8. Our method achieves competitive performance (75.00%) compared to the upper-bound accuracy from joint training.

Furthermore, we report the mAP evolution during incremental training in Figure 9(a). The activation regularization methods (*e.g.*, L2 loss) outperform the network parameters regularization methods (*e.g.*, EWC). Moreover, we visualize the Gram matrices of three methods. As the training proceeds, their differences with respect to the reference Gram matrices are maximized. Namely, the bright area in the three methods

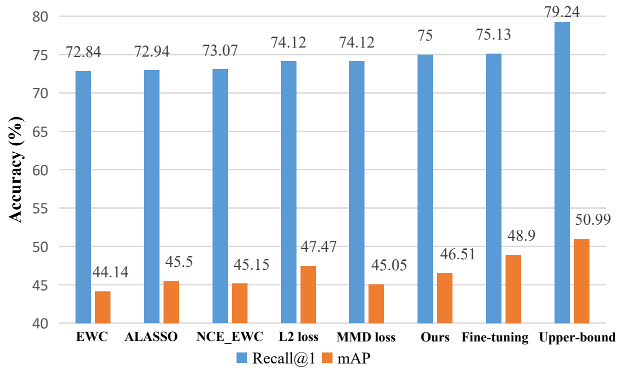

Fig. 8. The accuracy gain for the new classes on the CUB-Birds dataset. “Upper bound” accuracy is achieved by joint training.

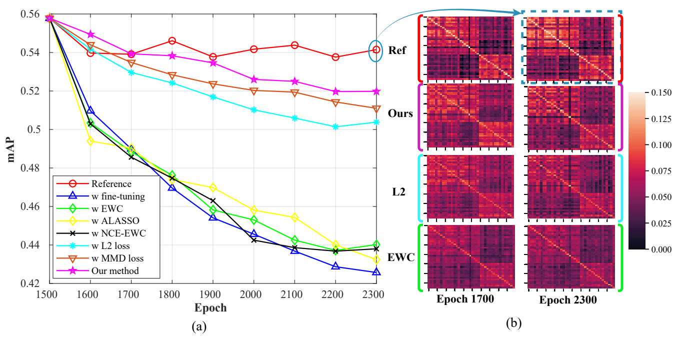

Fig. 9. (a) mAP evolution of old classes *(1-100)* tested on the CUB-Birds dataset under one-task scenario. (b) The Gram matrices of four representative methods (best viewed in color). More brightness indicates higher semantic correlations between two samples. The reference performance is obtained by joint training. Our method retains most semantics (higher brightness) compared to EWC and L2 loss.

becomes ambiguous. However, our method retains most semantics of old classes (more brightness) than the other two continual learning strategies even at the last training epoch.

D. Multi-task scenario evaluation

Multi-task scenario refers to the case that m new classes are divided evenly into several groups and added sequentially. For the CUB-Birds dataset, the remaining 100 new classes are split into 4 disjoint groups, with 25 classes per group; For the Stanford-Dogs dataset, we also get 4 groups with 15 classes per group. Thus, there are 4 steps incremental training for each dataset. For each step, the model is trained only on the images from a new class group (*e.g.*, *classes (126-150)* of the CUB dataset) and is tested separately in prior groups (*e.g.*, *classes (1-100)* and *classes (101-125)*) to evaluate the forgetting rate of this step. Note that incremental performance is *insensitive* to the arrival order and choice of new classes since the tasks do not depend on softmax-based probabilities [56].

Accuracy change range. We estimate the features of previous models (using Eqs. 7 and 8) based on the accuracy change defined in Eq. 6. Concretely, we use mAP to calculate the accuracy range. For instance, on the CUB-Birds dataset, model f_0*(1-100)* takes as input the first group of new classes (see Figure 3) and produces an incrementally-trained model f_1*(101-125)*. In terms of mAP, it degrades from 54.20% to 52.44% on the $n = 100$ old classes while increases from 29.82% to 52.27% on the m=25 new classes. These recorded mAPs are used to calculate the accuracy change range $(\alpha_{drop}, \alpha_{gain})$ using Eq. 6. Finally, the mAP change range is (-0.0325, 0.7528) during task $t = 1$ and is used to estimate the features for model f_0 when training the next task $t = 2$, without storing this model. The estimated features serve as an extra regularization for training task $t = 2$ in which the knowledge is mainly transferred from the model f_1*(101-125)* to f_2*(126-150)*. This process is performed repeatedly until all new class groups are added. The earlier feature estimation procedure becomes less reliable as more groups of new classes are added. We solve this issue by decreasing importance factors in Eq. 11. Moreover, we demonstrate the efficacy of stability factor λ_2 in Table II, which is kept the same in the multi-task scenario.

We adopt forgetting measurement [48] to quantify the forgetting ratio. Specifically, the forgetting ratio for a particular task is defined as the difference between the maximum accuracy gained throughout the incremental training process in the past and the accuracy the currently-trained model has, then all t tasks forgetting ratios are averaged:

$$forgetting = \frac{1}{t-1}\sum_{j=1}^{t-1}\Bigl(\max_{l\in\{1,\dots,t-1\}} Acc_{l,j} - Acc_{t,j}\Bigr), \forall j < t \tag{12}$$

where $Acc_{t,j}$ denotes the accuracy of j^{th} group of new classes evaluated by the model trained on the task t. Concretely, we employ the mAP metric as Acc for evaluation. When the model has been incrementally trained up to task t, we measure and then average all previous forgetting ratios $(1, 2, ..., t-1)$ using Eq. 12 as final forgetting evaluation.

The average forgetting ratios are depicted in Figure 10. Note that we use the task index to indicate the group of new classes being added. For example, “$t = 2$” on the CUB-Birds dataset means the model is training on the 2^{nd} group of new classes and then tested on *classes (1-100)* and *classes (101-125)* separately. Obviously, all methods suffer catastrophic forgetting on two datasets. In particular, fine-tuning on a new task leads to significant forgetting on the old tasks. EWC and ALASSO cannot reduce the forgetting issue ideally in the multi-task scenario. By contrast, activation regularization methods perform better on two datasets. Particularly, MMD loss and our method, by distilling feature correlations, can significantly reduce the forgetting ratio compared to the L2-regularized feature alignment method. Our method can further largely mitigate the forgetting ratio when feature estimation is considered into correlations distillation. Finally, our method has the least forgetting ratio (up to 10%) on these two datasets.

After all new tasks are added sequentially (*i.e.* $t = 4$), we get the final model f_4*(176-200)* or f_4*(106-120)* for this task. We measure the accuracy of each prior task (*i.e.*, class group) using the final model. We take Recall@1 as a metric for demonstration, as shown in Figure 11, including the performance for the previous tasks and the last new task. In this experiment, we use the performance of joint training as reference upper bound. In terms of Recall rate tested on the last new class group (*i.e.*, *classes (176-200)* and *classes (106-*

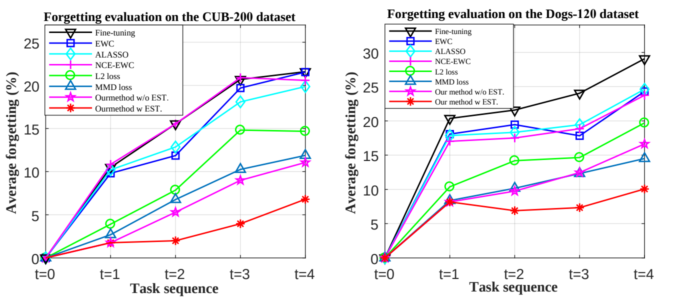
Fig. 10. Average forgetting evaluation. "w/o EST." indicates that feature ESTimation strategy is not included in our method (see Eq. 11). The forgetting is measured on previous old classes after training on current new classes. The forgetting ratios over all previous tasks are averaged to show. The higher value indicates the more severe forgetting.

120)), we find all six incremental learning algorithms and the fine-tuning method (without any knowledge distillation) have similar performance, close to the upper bound, especially for the Stanford-Dogs dataset. However, in terms of Recall on previous tasks, feature correlations used as knowledge can lead to a better-performing performance than other counterparts, closer to the upper bound, which means that our method suffers less forgetting on these preceding tasks. For instance, when tested the final model f_4*(176-200)* on the old *classes (1-100)* of the CUB-Birds dataset, our method achieves around 73% of Recall@1, 7% lower than the upper bound (80%), whereas other methods achieve less than 70%.

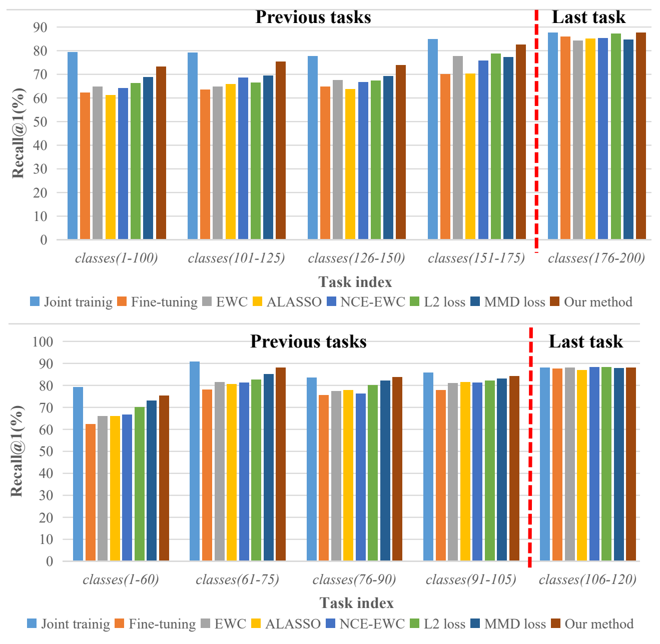
Fig. 11. The Recall@1 evaluation of each task (class group) at the end of the 4-step incremental learning. For instance, the model f_4*(176-200)* incrementally-trained on 4th new *classes (176-200)* at task $t = 4$ and is tested on all previously seen class groups. (a) Tested on the CUB-Birds dataset; (b) Tested on the Stanford-Dogs dataset.

We have demonstrated that our method can reduce the catastrophic forgetting on the previous tasks effectively. Also, the performance of the new task is essential to evaluate. As the incremental training proceeds, we report the Recall@1 on the new task during each incremental step in Figure 12. That is, we record the accuracy of new classes every time these classes are added. The results illustrate the evolution of performance on new classes. Obviously, we observe that all methods have similar Recall evolution and their performance is close to each other, especially for the Stanford-Dogs dataset.

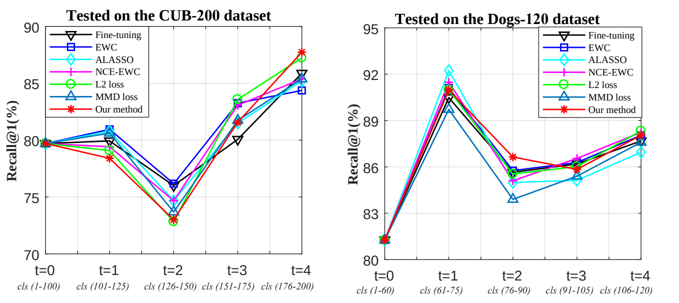
Fig. 12. The Recall@1 evolution tested on each new incoming class group

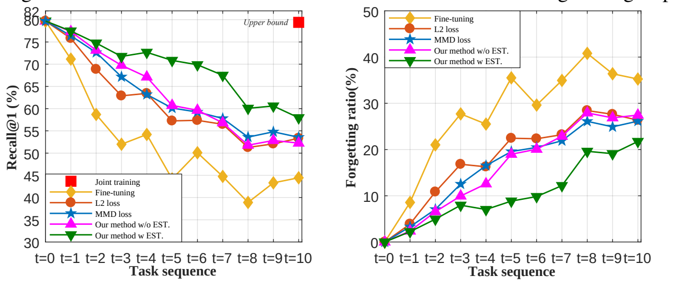
Fig. 13. 10-task performance comparison on the old *classes (1-100)*. The testing model is trained at the end of 10 tasks sequence on CUB-Birds. (a) Evolution of Recall@1; (b) Forgetting ratio evaluated on Recall@1.

We evaluate the case when more tasks are added sequentially on the CUB-Birds dataset. Concretely, the remaining $m = 100$ new classes are divided into 10 groups evenly. We focus on activation regularization algorithms and compare with L2 and MMD loss regularized methods. After the final model f_{10}*(191-200)* is trained at the end of the task sequence (*i.e.*, new *classes (191-200)*), we test this model on the original *classes (1-100)*, which suffer the most severe forgetting. The results are reported in Figure 13. Obviously, on the original *classes (1-100)*, correlations distillation with feature estimation method reduces the forgetting on *classes (1-100)* effectively.

E. Ablation study

(1) Efficacy of feature estimation

Feature estimation is introduced in Eq. 11 to reduce forgetting in the multi-task scenario. Here, we explore the efficacy of feature estimation. For this purpose, we consider a vanilla correlations distillation only from task $(t-1)$ to task t, *i.e.*, without using the feature estimation. Therefore, the loss for training is $L = \lambda_1 L_{triplet} + \lambda_2 L_{corr}^{(t-1)\to t}$.

We follow previous experimental protocols and conduct this study on the CUB-Birds dataset. We depict the Recall@1 and mAP evolution in Figure 14. Note that it is unnecessary to estimate feature drifts when task $t = 1$. When more new classes are added, distilling as knowledge feature correlations like MMD loss and our vanilla distillation method is more effective than L2 loss for reducing performance degradation. Also, vanilla distillation without feature estimation has a higher performance than MMD loss. When feature estimation strategy

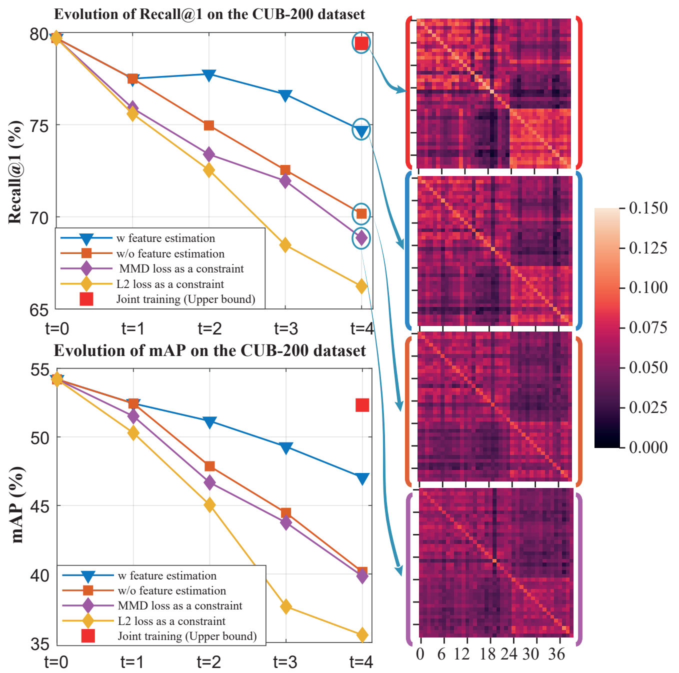

Fig. 14. Efficacy exploration for (a) Recall@1 and (b) mAP evolution *only* tested on the original *classes (1-100)*. We show the correlation matrices at the end of incremental training. This visualization further indicates that learning with feature estimation makes its performance closer to the upper bound.

is used, additional regularization from unsaved models can effectively retain more previously-learned knowledge, thereby leading to less forgetting on the original *classes (1-100)*.

(2) Influence of hyper-parameter

We show the efficacy of feature estimation in Figure 14. However, it seems that the estimated features in Eq. 11 act as augmented components for reducing catastrophic forgetting. In other words, the forgetting ratio reducing on the old classes might be realized by the hyper-parameter. To this end, we explore the influence of hyper-parameter. Following previous experimental protocols, we consider two-step incremental training on the CUB-Birds dataset where only new *classes (101-125)* and *classes (126-150)* are sequentially added. We do not consider task $t = 1$ is because there is no feature estimation in this task. When new *classes (126-150)* are adding at task t=2, the deep network is trained, using Eqs. 3 and 11, under four conditions: **case (a)** without feature estimation, **case (b)** with hyper-parameter augmented, **case (c)** with feature estimation, and **case (d)** with two-model distillation. The case (a) is viewed as a baseline where the correlations are distilled only from the penultimate model f_1*(101-125)* to the to-be-trained model f_2*(126-150)* by using their actual features. The case (d) is a complete method in which the previous models f_0*(1-100)* and f_1*(101-125)* are both saved for regularizing the training of current task $t = 2$. In contrast, it is unnecessary for our method (case (c)) to save the model f_0*(1-100)*.

The results are reported in Table IV. Naturally, the complete method in the case (d) produces an optimal performance on the old classes because all models are available. In terms of the baseline method, due to no distillation regularization, the trained model f_2*(126-150)* has the best performance on the new classes. For instance, its mAP reaches the maximal 52.45%. However, this model degrades performance heavily on the old classes to a minimal mAP (48.09%). In contrast, when the hyper-parameter of the baseline is augmented from λ_2 to $\lambda_2(1+\frac{1}{(t-1)})$. The trained model f_2*(126-150)* reduces forgetting on the old classes but limits the learning on the new classes. In particular, compared to the baseline method, the mAP of the case (b) on the old *classes (1-100)* reaches a maximal 50.71%, while it has the lowest Recall@1 (75.00%) and mAP (50.87%) on the new *classes (126-150)*. Therefore, Simply increasing the hyper-parameter of the stability term λ_2 in Eq. 3 cannot tackle well the stability-plasticity dilemma on the old tasks and new task because no extra knowledge is transferred. By contrast, training by using the feature estimation method can achieve competitive accuracy, taking both the old classes and new classes into account. Specifically, the model trained using the feature estimation method has a similar performance to the "two-model distillation" method on the old classes (76.19% $\rightarrow$ 76.91% of Recall@1). Meanwhile, the performance on the new classes is close to that of the baseline method (76.33% $\rightarrow$ 76.83% of Recall@1).

F. Comparison with image classification

Incremental learning has been widely explored in image classification tasks [17][6][18]. We claim that incremental image retrieval is more challenging. As noted, image classification tasks concentrate on the classification probabilities, while feature matching is the core for image retrieval tasks. Thus, a small change on the retrieval features would produce significant impact on the matching between features. To demonstrate this, we build an additional classifier on the top of retrieval features, and further train the deep network by using the popular LwF method [37] under the one-task scenario on CUB-Birds. During testing, we add Gaussian noise, sampling from the distribution $\mathcal{N}(0, 0.1)$, to all testing images, which affect the retrieval features and the final classification probabilities. We vary the ratio of Gaussian noise from 0 to 100%. We consider the evolution of forgetting degradation for these two tasks. The results are reported in Figure 16. As a result, image retrieval task suffers from more serious forgetting than image classification task with the same distraction noise.

G. Retrieval visualization

We visualize the retrieval results in FGIR by using different methods on the CUB-Birds dataset. We use the model trained at the end of the 4-step sequentially incremental training, *i.e.*, the model f_4*(176-200)*, and test this model on the old *classes (1-100)*. Considering the differences among images are subtle, we report the retrieved images and corresponding class names. The top 6 retrieved results are shown in Figure 15. For other tasks, their results are reported in the *supplemental materials.*

We select an image from class "*Pied Billed Grebe*" as the query item. This image is difficult to retrieve and is prone to cause forgetting issue because the color of the object in this image is similar to the background, as well as its incomplete appearance. Overall, all methods can return images with similar scenes. Other incremental algorithms suffer catastrophic forgetting and return more incorrect images. By contrast, our method effectively reduces the forgetting ratio and still returns

TABLE IV
HYPER-PARAMETER EXPLORATION RESULTS (%) ON THE CUB-BIRDS DATASET WHERE TRAINING TASK $t = 2$. WE SET λ_1 = 1 AND λ_2 = 10 IN THIS ABLATION STUDY. L_{actual} MEANS THAT THE LOSS TERM IS COMPUTED BY USING ACTUAL FEATURES, WHEREAS $L_{virtual}$ DENOTES THAT THE LOSS TERM IS COMPUTED BY USING ESTIMATED VIRTUAL FEATURES.

Configurations			Old *classes (1-100)*		New *classes (126-150)*	
Experiment conditions		The form of loss function $L =$	Recall@1	mAP	Recall@1	mAP
Case (a)	Baseline	$\lambda_1 L_{triplet} + \lambda_2 \left(L_{actual}^{(t-1)\to t}\right)$	74.75	48.09	76.83	52.45
Case (b)	Hyper-parameter-augmented	$\lambda_1 L_{triplet} + \lambda_2 \left(L_{actual}^{(t-1)\to t} + \frac{1}{(t-1)} L_{actual}^{(t-1)\to t}\right)$	76.69	50.71	75.00	50.87
Case (c)	With feature estimation	$\lambda_1 L_{triplet} + \lambda_2 \left(L_{actual}^{(t-1)\to t} + \frac{1}{(t-1)} L_{virtual}^{(t-2)\to t}\right)$	76.19	50.45	76.33	51.79
Case (d)	Two-model distillation [11]	$\lambda_1 L_{triplet} + \lambda_2 \left(L_{actual}^{(t-1)\to t} + L_{actual}^{(t-2)\to t}\right)$	76.61	50.49	76.50	51.92

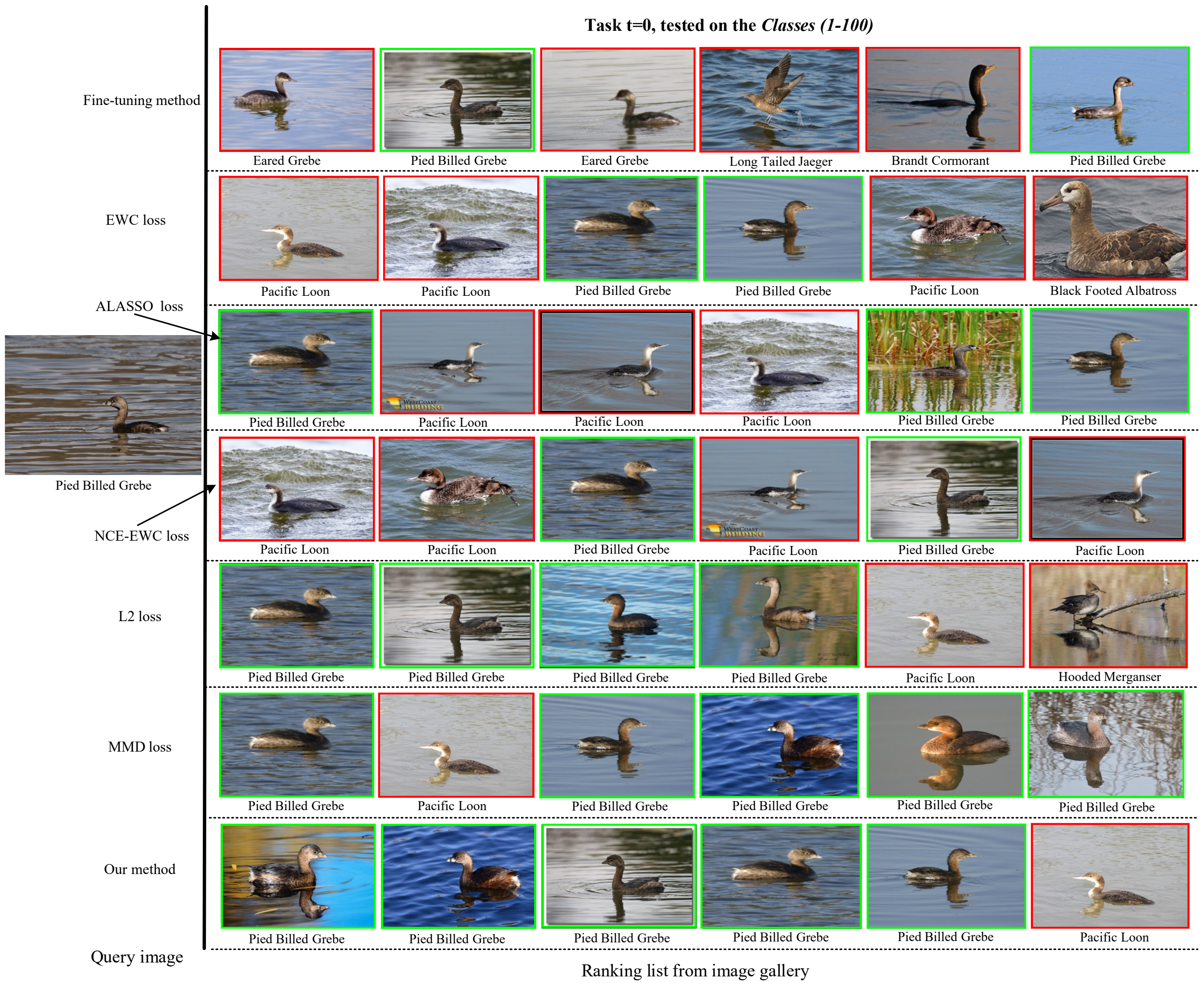

Fig. 15. Visualization of retrieved images and their class names on the CUB-Birds dataset. The Top 6 images tested on *classes (1-100)* are listed from left to right. For all methods, the query image is the same. The red box means an image is retrieved incorrectly, while the green box indicates the retrieved image has the same class label as the query image.

more correct images of the old tasks after a process of 4-step incremental learning.

H. Limitations and potential solutions

Although our method has achieved promising results in these experiments, it still shows several limitations as follows:

First, effectively estimating representations for all previous models depends on the parameter inheritance of model initialization at the start of each incremental step. However, estimated features from the penultimate model to the first one are not accurate enough due to the accumulative estimation errors. We resolved this limitation by aligning estimated features with descending importance and demonstrated its effectiveness ex-

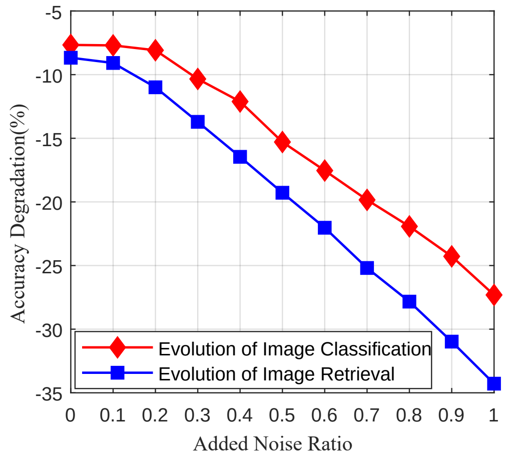

Fig. 16. Evolution for Incremental image classification and incremental image retrieval with respect to forgetting ratio.

perimentally. Nevertheless, distilling knowledge on the stream of models is worth further investigation theoretically. Sequence modelling via the recurrent network [57] may be a direction deserved to be explored.

Second, we focused on the representations extracted from the teacher-student structure to distill correlations. Thus, both old tasks and new tasks are trained on the same representations. However, regularizing directly on the representations may be overly restrictive for the learning on the new tasks. We find the accuracy of new tasks on the CUB-Birds dataset is still lower than the upper bound of joint training, see Table III. For this limitation, instead of regularizing the representations, it may be promising to project them into a sub-space using an auto-encoder or a variational auto-encoder. Afterwards, informative parts of the representations for the old tasks are captured and kept unchanged, while others that are not meaningful for the old tasks allow the learning for new tasks.

Third, we only evaluated our method on fine-grained datasets where training images share similar semantic commonalities, which is beneficial for reducing catastrophic forgetting. Despite its effectiveness, it is still far from the practical scenario where data from old tasks and new tasks are heterogeneous. Therefore, additional regularization terms are needed to examine for the heterogeneous data.

V. Conclusion

In this work, we explored fine-grained image retrieval in the context of incremental learning, where one-task and multi-task scenarios are validated. To achieve a trade-off performance for old tasks and new tasks, we used new data only and regularized their features extracted from the teacher model and the student model. In terms of multi-task incremental learning, saving all previous models for correlations distillation may cause a great demand in memory storage. We made an attempt to address the issue via a feature estimation method. That is, instead of storing a stream of old models, we saved the accuracy of models to compute the accuracy change during training each task. The semantic correlations of the estimated features, as an additional regularization, further mitigated the catastrophic forgetting ratio on previous tasks. Compared to previous approaches, the advantages of the proposed method were verified by thorough quantitative and qualitative results on two fine-grained datasets. Now, incremental image retrieval methods still need supervisory information. In the future, it is potentially valuable to explore incremental image retrieval in an unsupervised learning manner. Further, the data used in old tasks and new tasks share similar semantic commonalities, it is also interesting to examine for heterogeneous data.

Acknowledgment

This work was supported by LIACS MediaLab at Leiden University, China Scholarship Council (CSC No. 201703170183), and the National Natural Science Foundation of China under Grant 71701205, 62022091. We would like to thank NVIDIA for the donation of GPU cards.

References

[1] M. McCloskey and N. J. Cohen, “Catastrophic interference in connectionist networks: The sequential learning problem,” in *Psychology of learning and motivation*, 1989, vol. 24, pp. 109–165.
[2] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, “Large scale incremental learning,” in *CVPR*, 2019, pp. 374–382.
[3] D. Lopez-Paz and M. Ranzato, “Gradient episodic memory for continual learning,” in *NeurIPS*, 2017, pp. 6467–6476.
[4] H. Shin, J. K. Lee, J. Kim, and J. Kim, “Continual learning with deep generative replay,” in *NeurIPS*, 2017, pp. 2990–2999.
[5] G. M. van de Ven and A. S. Tolias, “Generative replay with feedback connections as a general strategy for continual learning,” *arXiv preprint arXiv:1809.10635*, 2018.
[6] X. Yao, T. Huang, C. Wu, R.-X. Zhang, and L. Sun, “Adversarial feature alignment: Avoid catastrophic forgetting in incremental task lifelong learning,” *Neural computation*, vol. 31, no. 11, pp. 2266–2291, 2019.
[7] K. Parshotam and M. Kilickaya, “Continual learning of object instances,” in *CVPR Workshops*, 2020, pp. 224–225.
[8] W. Chen, Y. Liu, W. Wang, T. Tuytelaars, E. M. Bakker, and M. Lew, “On the exploration of incremental learning for fine-grained image retrieval,” in *BMVC*, 2020, pp. 1–10.
[9] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
[10] L. Xie, J. Wang, B. Zhang, and Q. Tian, “Fine-grained image search,” *IEEE Transactions on Multimedia*, vol. 17, no. 5, pp. 636–647, 2015.
[11] P. Zhou, L. Mai, J. Zhang, N. Xu, Z. Wu, and L. S. Davis, “M2kd: Multi-model and multi-level knowledge distillation for incremental learning,” in *BMVC*, 2020, pp. 1–10.
[12] S. Hou, X. Pan, C. Change Loy, Z. Wang, and D. Lin, “Lifelong learning via progressive distillation and retrospection,” in *ECCV*, 2018, pp. 437–452.
[13] C. Wu, L. Herranz, X. Liu, J. van de Weijer, B. Raducanu, *et al.*, “Memory replay gans: Learning to generate new categories without forgetting,” in *NeurIPS*, 2018, pp. 5962–5972.
[14] H. Jung, J. Ju, M. Jung, and J. Kim, “Less-forgetful learning for domain expansion in deep neural networks,” in *AAAI*, 2018, pp. 3358–3364.
[15] W. Park, D. Kim, Y. Lu, and M. Cho, “Relational knowledge distillation,” in *CVPR*, 2019, pp. 3967–3976.
[16] D. Park, S. Hong, B. Han, and K. M. Lee, “Continual learning by asymmetric loss approximation with single-side overestimation,” in *ICCV*, 2019, pp. 3335–3344.
[17] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, “Overcoming catastrophic forgetting in neural networks,” *PNAS*, vol. 114, no. 13, pp. 3521–3526, 2017.
[18] Y. Wang, X. Fan, Z. Luo, T. Wang, M. Min, and J. Luo, “Fast online incremental learning on mixture streaming data,” in *AAAI*, 2017, pp. 2739–2745.
[19] K. Shmelkov, C. Schmid, and K. Alahari, “Incremental learning of object detectors without catastrophic forgetting,” in *ICCV*, 2017, pp. 3400–3409.
[20] R. Del Chiaro, B. Twardowski, A. Bagdanov, and J. van de Weijer, “Ratt: Recurrent attention to transient tasks for continual image captioning,” *NeurIPS*, vol. 33, 2020.

[21] U. Michieli and P. Zanuttigh, "Incremental learning techniques for semantic segmentation," in *ICCV Workshops*, 2019.

[22] W. W. Ng, X. Tian, W. Pedrycz, X. Wang, and D. S. Yeung, "Incremental hash-bit learning for semantic image retrieval in nonstationary environments," *IEEE transactions on cybernetics*, vol. 49, no. 11, pp. 3844–3858, 2018.

[23] X. Tian, W. Ng, H. Wang, and S. Kwong, "Complementary incremental hashing with query-adaptive re-ranking for image retrieval," *IEEE Transactions on Multimedia*, pp. 1–15, 2020.

[24] D. Wu, Q. Dai, J. Liu, B. Li, and W. Wang, "Deep incremental hashing network for efficient image retrieval," in *CVPR*, 2019, pp. 9069–9077.

[25] D. Mandal, Y. Annadani, and S. Biswas, "Growbit: Incremental hashing for cross-modal retrieval," in *ACCV*. Springer, 2018, pp. 305–321.

[26] J. Qi, Y. Peng, and Y. Zhuo, "Life-long cross-media correlation learning," in *ACM MM*, 2018, pp. 528–536.

[27] D. Mandal and S. Biswas, "A novel incremental cross-modal hashing approach," *arXiv preprint arXiv:2002.00677*, 2020.

[28] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, pp. 1–31, 2021.

[29] M. Zhai, L. Chen, F. Tung, J. He, M. Nawhal, and G. Mori, "Lifelong gan: Continual learning for conditional image generation," in *ICCV*, 2019, pp. 2759–2768.

[30] M. Yuan and Y. Peng, "Ckd: Cross-task knowledge distillation for text-to-image synthesis," *IEEE Transactions on Multimedia*, vol. 22, no. 8, pp. 1955–1968, 2019.

[31] L. Yu, V. O. Yazici, X. Liu, J. v. d. Weijer, Y. Cheng, and A. Ramisa, "Learning metrics from teachers: Compact networks for image embedding," in *CVPR*, 2019, pp. 2907–2916.

[32] B. Zhang, D. Xiong, J. Su, and J. Luo, "Future-aware knowledge distillation for neural machine translation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2278–2287, 2019.

[33] Z. Yu, L. Chen, Z. Cheng, and J. Luo, "Transmatch: A transfer-learning scheme for semi-supervised few-shot learning," in *CVPR*, 2020, pp. 12 856–12 864.

[34] H.-T. Li, S.-C. Lin, C.-Y. Chen, and C.-K. Chiang, "Layer-level knowledge distillation for deep neural network learning," *Applied Sciences*, vol. 9, no. 10, p. 1966, 2019.

[35] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *CVPR*, 2017, pp. 4133–4141.

[36] Y. Chebotar and A. Waters, "Distilling knowledge from ensembles of neural networks for speech recognition." in *Interspeech*, 2016, pp. 3439–3443.

[37] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[38] X. Huang and Y. Peng, "Tpckt: two-level progressive cross-media knowledge transfer," *IEEE Transactions on Multimedia*, vol. 21, no. 11, pp. 2850–2862, 2019.

[39] K. Lee, K. Lee, J. Shin, and H. Lee, "Overcoming catastrophic forgetting with unlabeled data in the wild," in *ICCV*, 2019, pp. 312–321.

[40] X. Ma, T. Zhang, and C. Xu, "Multi-level correlation adversarial hashing for cross-modal retrieval," *IEEE Transactions on Multimedia*, vol. 22, no. 12, pp. 3101–3114, 2020.

[41] Y. Peng and J. Qi, "Show and tell in the loop: Cross-modal circular correlation learning," *IEEE Transactions on Multimedia*, vol. 21, no. 6, pp. 1538–1550, 2018.

[42] Y. Peng, J. Qi, X. Huang, and Y. Yuan, "Ccl: Cross-modal correlation learning with multigrained fusion by hierarchical network," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 405–420, 2017.

[43] Z. Li, J. Tang, and T. Mei, "Deep collaborative embedding for social image understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2070–2083, 2018.

[44] Z. Peng, Z. Li, J. Zhang, Y. Li, G.-J. Qi, and J. Tang, "Few-shot image recognition with knowledge transfer," in *ICCV*, 2019, pp. 441–449.

[45] B. Peng, X. Jin, J. Liu, D. Li, Y. Wu, Y. Liu, S. Zhou, and Z. Zhang, "Correlation congruence for knowledge distillation," in *ICCV*, 2019, pp. 5007–5016.

[46] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," in *ICCV*, 2019, pp. 1365–1374.

[47] W. Li, J. Huo, Y. Shi, Y. Gao, L. Wang, and J. Luo, "Online deep metric learning," *arXiv preprint arXiv:1805.05510*, 2018.

[48] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *ECCV*, 2018, pp. 532–547.

[49] P. Upchurch, J. Gardner, G. Pleiss, R. Pless, N. Snavely, K. Bala, and K. Weinberger, "Deep feature interpolation for image content changes," in *CVPR*, 2017, pp. 7064–7073.

[50] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[51] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization: Stanford dogs," in *CVPR Workshop*, vol. 2, no. 1, 2011.

[52] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, "Multi-similarity loss with general pair weighting for deep metric learning," in *CVPR*, 2019, pp. 5022–5030.

[53] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *CVPR*, 2016, pp. 4004–4012.

[54] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2010.

[55] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *CVPR*, 2017, pp. 3366–3375.

[56] J.-M. Perez-Rua, X. Zhu, T. M. Hospedales, and T. Xiang, "Incremental few-shot object detection," in *CVPR*, 2020, pp. 13 846–13 855.

[57] C. Fu, W. Pei, Q. Cao, C. Zhang, Y. Zhao, X. Shen, and Y.-W. Tai, "Non-local recurrent neural memory for supervised sequence modeling," in *CVPR*, 2019, pp. 6311–6320.