Lifelong Learning via Progressive Distillation and Retrospection

Saihui Hou[1*][0000−0003−4689−2860], Xinyu Pan[2*][0000−0001−9951−4842], Chen Change Loy[3][0000−0001−5345−1591], Zilei Wang[1][0000−0003−1822−3731], and Dahua Lin[2][0000−0002−8865−7896]

[1] Department of Automation, University of Science and Technology of China
[2] Department of Information Engineering, The Chinese University of Hong Kong
[3] Nanyang Technological University

Abstract. Lifelong learning aims at adapting a learned model to new tasks while retaining the knowledge gained earlier. A key challenge for lifelong learning is how to strike a balance between the *preservation* on old tasks and the *adaptation* to a new one within a given model. Approaches that combine both objectives in training have been explored in previous works. Yet the performance still suffers from considerable degradation in a long sequence of tasks. In this work, we propose a novel approach to lifelong learning, which tries to seek a better balance between *preservation* and *adaptation* via two techniques: *Distillation* and *Retrospection*. Specifically, the target model adapts to the new task by knowledge distillation from an intermediate expert, while the previous knowledge is more effectively preserved by caching a small subset of data for old tasks. The combination of *Distillation* and *Retrospection* leads to a more gentle learning curve for the target model, and extensive experiments demonstrate that our approach can bring consistent improvements on both old and new tasks[4].

Keywords: Lifelong Learning, Knowledge Distillation, Retrospection.

1 Introduction

Lifelong learning aims at adapting a learned model to new tasks while retaining the knowledge acquired in the past. With the wide adoption of computer vision in real-world applications, there is an increasing demand for learning systems that are able to carry out lifelong learning over a series of tasks in a continual fashion. For example, a real-world object classification system is often required to be upgraded constantly by absorbing the knowledge from fresh domains. Directly repeating the training process with both previous and new data is often infeasible, due to various issues such as computation cost, storage budget, and privacy. For lifelong learning, a key challenge is to overcome the risk of catastrophic forgetting [9], namely a learned model usually suffers from accuracy degradation on old tasks when it adapts to a new one.

[4] Project page: http://mmlab.ie.cuhk.edu.hk/projects/lifelong/
[*] indicates joint first authorship.

In this work, we focus on incremental multi-task object categorization in the context of deep learning. Here we assume that the classification tasks for different domains arrive in a sequential manner and a single model is required to perform well on all presented tasks at the end of each training stage. This setting serves as a reasonable starting point for further generalization. Some classical methods can be applied to the setting but with significant drawbacks. Specifically, (a) *Feature Extraction* [7] is suboptimal for a new task with the feature extractor frozen; (b) *Finetuning* [8], which adapts the whole network to new data, leads to a dramatic performance drop on old tasks; (c) *Joint Training* [4] brings the excessive demand for data storage and increasing training cost.

To overcome these drawbacks, various methods have been proposed and can be roughly divided into two categories. The first one [15, 3, 20, 12, 21] is based on knowledge distillation [10] and uses a modified cross-entropy loss to maintain the performance on old tasks. These methods have been proven to be effective, however, the performance drops when the target model is exposed to a sequence of tasks drawn from different distributions. The second one [13, 27, 2] focuses on the model itself and tries to identify the importance of parameters for old tasks, which is used as the guidance for adaptation to the new task. However, it is difficult to design a metric to weight all the parameters such that the performance on old tasks can be preserved very well, especially in a long sequence of tasks.

In this work, we propose a novel approach for lifelong learning on visual tasks, by drawing wisdom from human learning. When a student studies at school, he needs to gradually learn the knowledge of various courses without forgetting those previously learned. It is usually more efficient for him to learn the knowledge from a great teacher than directly from a book or by repeatedly doing exercises. Besides, he has to review those that have been learned early from time to time in order to not forget. Motivated by these observations, an approach consisting of two techniques, *Distillation* and *Retrospection*, is proposed with the aims of striking a better balance between the performance preservation on old tasks and the adaptation to a new task.

Distillation provides a novel way to adapt to a new task, which is the abbreviation of our algorithm named *Adaptation by Distillation*. Instead of directly finetuning on new data, we first train an *Expert CNN* dedicated to the new task and then uses it as an *advisor* to guide the adaptation of target model, via knowledge distillation [10]. Whereas previous works [15, 20] that utilize knowledge distillation for lifelong learning primarily focus on preserving the knowledge obtained in the past, we find that the distillation-based learning from an *Expert CNN*, which offers soft supervision, leads to a more gentle learning curve for adaptation to the new task. As a result, the target model can adapt more s-moothly, thus achieving better performance on the new task as well as old tasks. *Retrospection* allows the target model to revisit the previous data from time to time, which emulates how we human beings try not to forget. It differs from *Joint Training* [4] where the data for all tasks are completely available. Instead, *Retrospection* requires only a small fraction of previous data to be reserved. Our study shows that even a very small subset of data from the past can help remarkably preserve the performance on earlier tasks, without incurring significant cost on computation and storage.

In summary, our contributions of this work mainly lie in three aspects: (1) We propose a new algorithm named *Adaptation by Distillation* for multi-task lifelong learning, which is expected to become a better practice to train a single model for a long sequence of tasks. (2) We explore the new setting in multi-task lifelong learning with a small subset of old data available and show that *Retrospection* is greatly helpful for the performance preservation on old tasks. (3) Extensive experiments demonstrate that *Distillation+Retrospection* can bring consistent improvements over the baselines and outperform *Learning without Forgetting* [15] by a large margin. For example, on ImageNet which comes as the first task in the five-task scenario, the final accuracy by *Distillation+Retrospection* exceeds the baseline [15] by more than 6%.

2 Related Work

Our method is built on the insights of multiple earlier works, not only for lifelong learning but also for other visual tasks. In the section, we summarize the most related ones to our work, which are comprised of the following three parts.

Multi-task Learning. The goal of multi-task lifelong learning is to train a single model which can predict well on multiple tasks, with the data for different tasks provided sequentially. It is at the intersection of multi-task learning and lifelong learning. Standard multi-task learning [4] is equivalent to *Joint Training* described in Section 1. The initial objective is to make use of the knowledge across different tasks (*i.e.*, so-called inductive bias [17]) to improve the accuracy on each individual task, which has the benefit of relaxing the number of required samples per task. However, the main drawback of this standard practice for lifelong learning is that it requires all the data for different tasks available. In this work, we first validate the effectiveness of *Distillation* without accessing the data for old tasks. And then we further explore the setting with *Retrospection*, *i.e.*, a small subset of data is reserved for old tasks.

Knowledge Distillation. Knowledge distillation is proposed by Hinton *et al.* [10], where knowledge is transferred from a large network or a network assembly to a small network for efficient deployment. The small network is trained using a modified cross-entropy loss (denoted by KD-Loss), which encourages the responses of the original and new network to be similar. This method is widely used to produce a network of different structure that approximates the original one, *e.g.*, Romero *et al.* [22] transfer to a deeper and thinner network for network compression, Chen *et al.* [5] generate a deeper and wider network for fast hyper-parameter exploration. For lifelong learning, Li *et al.* [15] propose an algorithm called *Learning without Forgetting* and first introduce knowledge distillation to preserve the performance on old tasks. KD-Loss [10] is adopted to mimic the output of the original network in the adaptation to a new task. Our work is also based on knowledge distillation [10]. The significant differences between our work and [15] lie in the learning for the new task. In our algorithm, the target model adapts to a new task by distilling the knowledge from an intermediate *Expert CNN*, which can facilitate the learning on the new task and is also beneficial to preserve the performance on old tasks. Besides, Tzeng *et al.* [24] dealing with domain adaptation use knowledge distillation to help the training on a new domain. However, the soft labels in [24] come from the model trained on a well-labeled source domain, and they do not need to consider maintaining the performance on the source domain. While in our algorithm, the soft labels are obtained by fine-tuning an *Expert CNN* on new data ignoring the constraint of preserving the performance on old tasks.

Lifelong Learning based on Knowledge Distillation. As stated in Section 1, recent works on lifelong learning can be roughly divided into two categories, one of which is based on knowledge distillation. Besides *Learning without Forgetting* [15], here we discuss other works of this category [3, 20, 12, 21]. Aljundi *et al.* [3] propose to train multiple networks on different tasks and take an auto-encoder to choose a network for each test sample, where the algorithm in [15] is adopted when the task relatedness is high. Rannen *et al.* [20] also introduce an auto-encoder but the goal is to help preserve the crucial features for old tasks. Jung *et al.* [12] propose to approximate the features of the original network for old tasks rather than the output of the last layer. The focuses of these works, especially [20, 12], are to more effectively preserve the performance for old tasks, while the accuracy on the new task is comparable or a little inferior to the baseline [15]. Differently, our algorithm can simultaneously improve the performance on both old and new tasks, and the *Distillation*-based learning for a new task as well as the *Retrospection* on old data can also be integrated with these methods. Besides, Rebuffi *et al.* [21] deal with multi-class lifelong learning. The main difference between multi-class and multi-task lifelong learning lies in the prediction step: a multi-class learner has to train a unified classifier that predicts correctly any of the observed class, while a multi-task learner can make use of multiple classifiers, each of which is evaluated only on the data from its own domain. In this work, we focus on multi-task lifelong learning. Rebuffi *et al.* [21] also keep some data for old classes, while we first explore the setting with a small subset of old data available in the multi-task scenario and get some different observations, which will be provided in Section 4.

3 Distillation and Retrospection

The approach proposed in this work is illustrated in Figure 1. The framework consists of two key components: *Distillation* and *Retrospection*. It deals with multi-task lifelong learning aiming at training a single Convolutional Neural Network (CNN) that can perform reasonably well on a variety of classification tasks. The training data and ground truth for each task are presented to the model in a sequential manner. In each phase, the model evolves to a new task

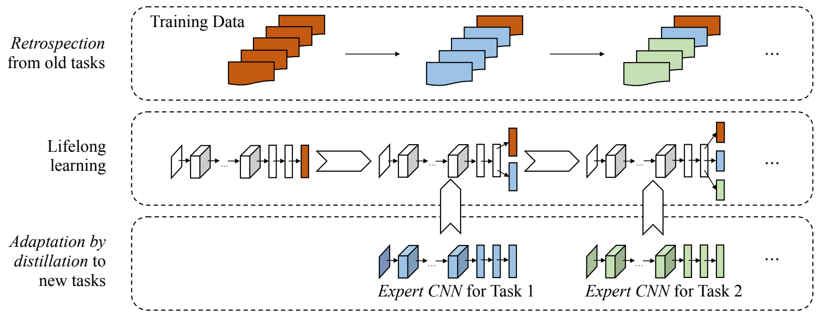


Fig. 1. Illustration of *Distillation* and *Retrospection*. The model learns the knowledge on a new task through the *Distillation* from an *Expert CNN*. *Retrospection* allows the model to revisit a small subset of data for old tasks.

without accessing all the data for old tasks. The input to our algorithm is an *Original CNN* that contains the feature extractor F and task-specific classifiers T_{o} for old tasks. The network learns to adapt to a new task by distilling the knowledge from an intermediate *Expert CNN*. The output is the updated feature extractor F^* and task-specific classifiers T_{o}^* for old tasks as well as a task-specific classifier T_{n}^* for the new task. In the following, we will first review *Learning without Forgetting* [15] as background. Then we will elaborate how *Distillation* works to facilitate the learning on the new task and simultaneously benefit the performance preservation on old tasks. Finally, we will introduce *Retrospection*, *i.e.*, a small subset of data is reserved for old tasks.

3.1 Background

Learning without Forgetting (LwF) [15] is a representative method for multi-task lifelong learning. It first introduces knowledge distillation [10] for lifelong learning to preserve the performance on old tasks. The loss function for adapting the model to a new task is the sum of two terms: L_{new}^F for the new task and L_{old}^F for the old task[5].

Specifically, in the context of image classification, L_{new}^F is the standard cross-entropy loss [14, 23]:

$$L_{\mathrm{new}}^F(X_{\mathrm{n}}, Y_{\mathrm{n}}) = -\frac{1}{|N_{\mathrm{n}}|} \sum_{i=1}^{|N_{\mathrm{n}}|} \sum_{k=1}^{K_{\mathrm{n}}} y_{\mathrm{n}}^{ik} \cdot \log\left(p_{\mathrm{n}}^{ik}\right), \tag{1}$$

where $X_{\mathrm{n}}/Y_{\mathrm{n}}$ are the training data and ground truth for the new task, N_{n} is a batch of samples drawn from X_{n}, K_{n} is the number of classes for the new task, y_{n}^i is the one-hot ground truth labels of the i-th sample, p_{n}^i is the corresponding

[5] The regularization terms are omitted for simplicity.

softmax output. The loss encourages the predictions of target model for the new task to match the one-hot labels.

L^F_{old} is the knowledge distillation loss (KD-Loss). In order to compute it, the output of *Original CNN* for the old task denoted by $\widehat{Y}_{\text{o}}$ is first computed and recorded before the training starts. It is worth noting that, since the data for the old task is not available [15], $\widehat{Y}_{\text{o}}$ is computed on new data. For image classification, $\widehat{Y}_{\text{o}}$ is the set of label probabilities, *i.e.*soft labels. Then, L^F_{old} is computed as follows in the training:

$$L^F_{\text{old}}(X_{\text{n}}, \widehat{Y}_{\text{o}}) = -\frac{1}{|N_{\text{n}}|} \sum_{i=1}^{|N_{\text{n}}|} \sum_{k=1}^{K_{\text{o}}} \hat{y}_{\text{o}}^{(ik)'} \cdot \log\left(p_{\text{o}}^{(ik)'}\right), \tag{2}$$

where K_{o} is the number of classes for the old task, $\hat{y}_{\text{o}}^{(i)'}$ and $p_{\text{o}}^{(i)'}$ are the modified versions of recorded soft labels by *Original CNN* and current network predictions for the old task:

$$\hat{y}_{\text{o}}^{(ik)'} = \frac{\left(\hat{y}_{\text{o}}^{(ik)}\right)^{1/\gamma_{\text{o}}}}{\sum_j \left(\hat{y}_{\text{o}}^{(ij)}\right)^{1/\gamma_{\text{o}}}},\ p_{\text{o}}^{(ik)'} = \frac{\left(p_{\text{o}}^{(ik)}\right)^{1/\gamma_{\text{o}}}}{\sum_j \left(p_{\text{o}}^{(ij)}\right)^{1/\gamma_{\text{o}}}}, \tag{3}$$

where γ_{o} is usually set to be greater than 1 which increases the weights of small values. L^F_{old} moves towards performance preservation by encouraging the current predictions for the old task to match the soft labels by *Original CNN*, though the predictions and soft labels are both computed on new data. When there are multiple old tasks, the loss in Eq. (2) is computed for each old task and then the sum of them is used for L^F_{old}.

There exist some limitations in *LwF*. First, for preserving the performance on old tasks, the target model adapts to a new task with the constraint of mimicking the output of *Original CNN* as much as possible. Though sometimes this constraint provides useful regularization to the cases with rare new samples, it is also likely to hinder the adaptation to the new task. Second, the performance on old tasks degrades a lot when the model is exposed to a long sequence of tasks for different domains [3] since the loss for old tasks is computed on the new coming data which is likely to be drawn from a significantly different distribution compared to the previous data.

3.2 Distillation

Distillation is the abbreviation of our algorithm named *Adaptation by Distillation*, which is motivated to facilitate the adaptation to a new task while preserving the performance on old tasks. The network structures are depicted in Figure 2. The main differences between *Distillation* and *LwF* lie in the learning on the new task. In our algorithm, the target model adapts to a new task through knowledge distillation instead of directly training on new data. The steps for *Distillation* are described below.

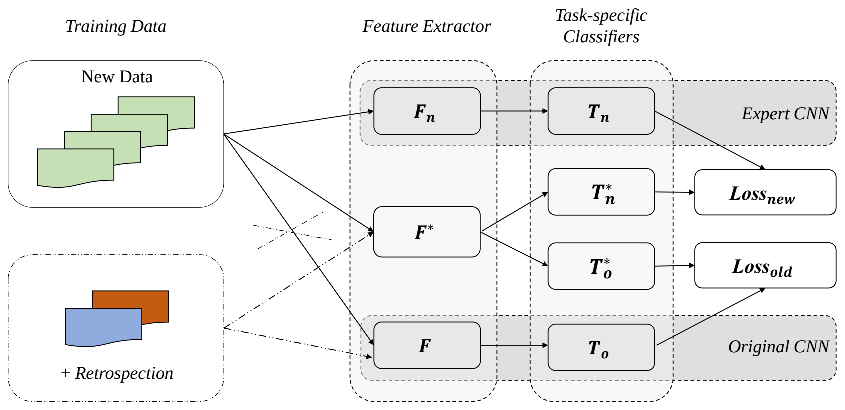


Fig. 2. Illustration of network structures for *Distillation+Retrospection*. The structures with or without *Retrospection* are the same, while the differences lie in the composition of training data and the computation of loss for old tasks. The responses of *Expert CNN* as well as *Original CNN* can be recorded before training, both of which do not bring additional GPU memory consumption. When adapting the original model to the new task, only weights of F^*, T_o^*, T_n^* are *not fixed*.

First, an *Expert CNN* is trained purely on the new task. The loss function has only one term for image classification, *i.e.*, the cross-entropy loss as in E-q. (1). The resulting *Expert CNN* is skilled at discriminating new data so that it theoretically provides an upper bound for the performance on the new task. Second, the responses of *Expert CNN* for new data denoted by $\widehat{Y}_\mathrm{n}$ are computed and recorded, which are used as the supervision for learning on the new task in the next step.

Finally, it comes to adapting to the new task without catastrophic forgetting. Knowledge distillation [10] is not only used to preserve the performance on old tasks, but also utilized for the adaptation to the new task. The loss function is also composed of two terms for the old and the new task respectively. The one for the old task is a type of KD-Loss computed as in Eq. (2), while the other for the new task is another KD-Loss instead of cross-entropy loss, which is computed as follows:

$$L_\mathrm{new}^D(X_\mathrm{n}, \widehat{Y}_\mathrm{n}) = -\frac{1}{|N_\mathrm{n}|}\sum_{i=1}^{|N_\mathrm{n}|}\sum_{k=1}^{K_\mathrm{n}} \hat{y}_\mathrm{n}^{(ik)'} \cdot \log\left(p_\mathrm{n}^{(ik)'}\right), \tag{4}$$

where $X_\mathrm{n}/\widehat{Y}_\mathrm{n}$ are the new training data and the soft labels output by *Expert CNN*, N_n is the batch drawn from X_n, K_n is the number of classes for the new task, $\hat{y}_\mathrm{n}^{(i)'}$ and $p_\mathrm{n}^{(i)'}$ are the modified versions of recorded soft labels by *Expert*

CNN and current network predictions for the new task:

$$\hat{y}_{\mathrm{n}}^{(ik)'} = \frac{\left(\hat{y}_{\mathrm{n}}^{(ik)}\right)^{1/\gamma_{\mathrm{n}}}}{\sum_j \left(\hat{y}_{\mathrm{n}}^{(ij)}\right)^{1/\gamma_{\mathrm{n}}}},\ p_{\mathrm{n}}^{(ik)'} = \frac{\left(p_{\mathrm{n}}^{(ik)}\right)^{1/\gamma_{\mathrm{n}}}}{\sum_j \left(p_{\mathrm{n}}^{(ij)}\right)^{1/\gamma_{\mathrm{n}}}}, \tag{5}$$

where γ_{n} is also set to be greater than 1 to enhance the contribution of small values. In other words, the learning of the new task is based on the knowledge distillation from *Expert CNN*. In the training, the one-hot labels of new data are replaced by the soft labels output by *Expert CNN*, which can enforce the relationship among classes [24] and thus facilitate the learning on the new task. Besides, *Distillation* is also beneficial for the performance preservation on old tasks since it is easier for *Original CNN* to match the output on new data to a soft distribution (*i.e.*soft labels by *Expert CNN*) instead of a very peaked one (*i.e.*one-hot labels).

3.3 Retrospection

Retrospection means that a small fraction of data for old tasks is reserved for lifelong learning. Though additional memory space is required, we find that a rather small subset of data for old tasks is enough to offer great help for performance preservation, especially with a long sequence of tasks for different domains, such as Scenes, Birds, Flowers and Aircrafts.

In comparison to *LwF*, the loss for the old task with *Retrospection* is computed in a different way:

$$L_{\mathrm{old}}^{R}(X_{\mathrm{o}}, \widehat{Y}_{\mathrm{o}}) = -\frac{1}{|N_{\mathrm{o}}|}\sum_{i=1}^{|N_{\mathrm{o}}|}\sum_{k=1}^{K_{\mathrm{o}}} \hat{y}_{\mathrm{o}}^{(ik)'} \cdot \log\left(p_{\mathrm{o}}^{(ik)'}\right), \tag{6}$$

where X_{o} is a small subset of data for the old task, Y_{o} is the recorded responses by *Original CNN*, N_{o} is a batch of samples drawn from X_{o}. Besides, $\hat{y}_{\mathrm{o}}^{(i)'}$ and $p_{\mathrm{o}}^{(ik)'}$ are the modified versions of recorded responses by *Original CNN* and current network predictions for the old task, which are computed as in Eq. (3).

Note that, with *Retrospection*, the loss for the old task (L_{old}^{R}) as well as the responses of *Original CNN* ($\widehat{Y}_{\mathrm{o}}$) is computed on the reserved small subset of data for the old task instead of new data. In the cases of more than one old tasks, the loss above is computed for each old task with the data from its own domain, and then L_{old}^{R} is computed as the sum of them in order to preserve the performance on these tasks.

3.4 Summary

In our approach consisting of *Distillation* and *Retrospection*, the loss function for lifelong learning is also composed of two terms for the old task and the new one respectively. Compared to *LwF*, *Distillation* adopts a type of KD-Loss instead

Table 1. The statistics of the datasets used in this work.

Task	Datasets	#Category	#Training	#Test
ImageNet	ILSVRC-2012 [19]	1000	1,281,167	50,000
Birds	CUB-200-2011 [26]	200	5994	5794
Flowers	Oxford Flowers [18]	102	2040	6149
Scenes	MIT Scenes [19]	67	5360	1340
Aircrafts	FGVC-Aircrafts [16]	100	6667	3333

of cross-entropy loss for the new task, while *Retrospection* updates the loss for the old task by computing it on the reserved small subset of data rather than on new data. In the final version of our approach, *i.e.*, *Distillation+Retrospection*, the loss function is the sum of L^D_{new} in Eq. (4) for the new task and L^R_{old} in Eq. (6) for the old task, which can help outperform *LwF* by a large margin on each individual task. Besides, *Distillation* is superior to *LwF* with or without *Retrospection*. *Retrospection* can also be integrated with *LwF* to significantly improve the performance on old tasks, which is also helpful for the new task.

4 Experiments

4.1 Settings

Experiments are conducted on a variety of classification tasks, including ImageNet [6], Scenes [19], Birds [26], Flowers [18] and Aircrafts [16]. We consider the sets of two and five tasks coming in a sequential manner.

Datasets. The statistics of the datasets evaluated in this work, which are also used in [15, 3, 20], are summarized in Table 1. For ImageNet [19], the evaluation is done on its validation set.

Implementation Details. All models are implemented with Caffe [11] and trained on Titan-X GPUs. AlexNet [14] is adopted as the backbone network due to its simplicity and efficient deployment, which is widely used in the literature for lifelong learning [15, 3, 20, 12, 27, 2]. The feature extractor F consists of the five convolutional layers and the first two fully connected layers, while the task-specific classifiers $T_{\text{o}}/T_{\text{n}}$ refer to the last fully connected layer. The standard practice described in [1] is applied to train the *Expert CNN*. As for the adaptation to the new task without catastrophic forgetting, it is conducted with the loss to mimic the output of *Original CNN* and we follow the similar practice. Stochastic gradient descent (SGD) is used for the optimization. The initial learning rate is set to 0.001 and scaled to its 1/10 three times until convergence. The training images (resized to 256×256) are randomly flipped and cropped as input, and no other data augmentation is used. The inference is done with a single center crop of test images. All the results are reported as the top-1 accuracy of percentage.

For other hyper-parameters, the batch sizes for $N_{\text{o}}/N_{\text{n}}$ are set to 128, and the temperatures for KD-Loss, *i.e.*, $\gamma_{\text{o}}/\gamma_{\text{n}}$, are set to 2. The loss weights for different

Table 2. Classification accuracy (%) for two-task scenario starting from ImageNet. *Feature Extraction* provides the reference performance for the first task while *Finetuning* provides the reference for the second one. *D* for *Distillation*, and *R* for *Retrospection*.

	ImageNet→Birds		ImageNet→Flowers		ImageNet→Scenes	
Feature Extraction	57.44 (ref)	50.12 (-7.07)	57.44 (ref)	83.10 (-3.99)	57.44 (ref)	60.22 (-2.61)
Finetuning	43.20 (-14.25)	57.19 (ref)	48.45 (-8.99)	87.09 (ref)	46.61 (-10.84)	62.84 (ref)
LwF [15]	54.49 (-2.95)	57.45 (+0.26)	55.77 (-1.67)	85.87 (-1.22)	55.01 (-2.43)	64.03 (+1.19)
D (**ours**)	55.34 (-2.11)	58.21 (+1.02)	55.95 (-1.49)	86.19 (-0.89)	55.65 (-1.79)	64.70 (+1.87)
LwF + R	55.61 (-1.83)	57.79 (+0.60)	56.48 (-0.96)	86.53 (-0.55)	55.71 (-1.73)	64.70 (+1.87)
D + R (**ours**)	**55.85 (-1.59)**	**59.55 (+2.36)**	**56.53 (-0.92)**	**87.02 (-0.07)**	**56.02 (-1.43)**	**65.00 (+2.16)**

tasks are set to 1, which turns out to be a reasonable choice in our implementation. For *Retrospection*, given that we deal with the cases that multiple tasks come in various sequences, the number of classes for old tasks rises at different rates. To enable a fair comparison, we choose to reserve five images for each class instead of a fixed budget for all classes. These images are randomly selected, and the strategy for *Retrospection* will be further discussed in Section 4.3.

Baselines. In order to validate the effectiveness of *Distillation* (denoted by D) and *Retrospection* (denoted by R), we compare our method to several baselines listed as follows:

(a) *Feature Extraction* [7]: as described in Section 1, it provides the reference performance for the first task.
(b) *Finetuning* [8]: as described in Section 1, it provides the reference performance for the last task.
(c) *Learning without Forgetting*(*LwF*) [15]: as described in Section 3.1, a representative method for multi-task lifelong learning.
(d) *Distillation*: as described in Section 3.2, *Adaptation by Distillation* without accessing the data for old tasks.
(e) *LwF+Retrospection*: the method to integrate *Retrospection* with *LwF*, L^R_{old} in Eq. (6) is adopted as the loss for old tasks instead of L^F_{old} in Eq. (2).
(f) *Distillation+Retrospection*: the final version of our approach, as described in Section 3.2 and Section 3.3, *Adaptation by Distillation* with a small subset of data reserved for old tasks.

Besides, the method from [20], denoted by *Encoder-based-LwF*, which is orthogonal to our approach, will be separately discussed and compared.

4.2 Performance Comparison

Two-task Scenario. Table 2 and Table 3 show the performance comparison in the two-task scenario. The experiments in Table 2 start from ImageNet, while those in Table 3 start from the smaller Flowers. In Table 3, ImageNet is not considered as a task in the sequence but used to pretrain the model for preventing training from scratch on small datasets.

Table 3. Classification accuracy (%) for two-task scenario starting from Flowers. *Feature Extraction* provides the reference performance for the first task while *Finetuning* provides the reference for the second one. *D* for *Distillation*, and *R* for *Retrospection.*

	Flowers→Birds		Flowers→Scenes		Flowers→Aircrafts	
Feature Extraction	87.09 (ref)	48.29 (-8.72)	87.09 (ref)	57.09 (-5.07)	87.09 (ref)	40.98 (-26.13)
Finetuning	72.97 (-14.12)	57.02 (ref)	72.97 (-14.12)	62.16 (ref)	70.88 (-16.20)	67.12 (ref)
LwF [15]	85.08 (-2.00)	54.55 (-2.46)	84.86 (-2.23)	61.87 (-0.30)	81.69 (-5.40)	66.10 (-1.02)
D (**ours**)	85.30 (-1.79)	56.64 (-0.38)	85.36 (-1.72)	62.31 (+0.15)	82.14 (-4.94)	67.57 (+0.45)
LwF + R	85.15 (-1.93)	56.79 (-0.22)	85.31 (-1.77)	62.54 (+0.37)	85.07 (-2.02)	66.88 (-0.24)
D + R (**ours**)	**85.38 (-1.71)**	**58.16 (+1.14)**	**85.73 (-1.35)**	**64.03 (+1.87)**	**85.57 (-1.51)**	**68.38 (+1.26)**

Table 4. Classification accuracy (%) for five-task scenario. The results are reported at the end of the last training stage. *LwF* is treated as the baseline here. *D* for *Distillation*, and *R* for *Retrospection.*

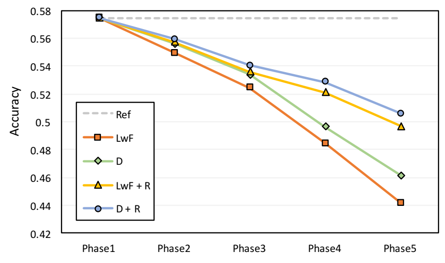
	Imagenet → Scenes → Birds → Flowers → Aircrafts					
	Imagenet	Scenes	Birds	Flowers	Aircrafts	Average
LwF [15]	44.20 (ref)	55.90 (ref)	52.22 (ref)	81.64 (ref)	65.80 (ref)	59.95 (ref)
D (**ours**)	46.15 (+1.95)	55.67 (-0.22)	53.17 (+0.95)	82.37 (+0.73)	66.79 (+0.99)	60.83 (+0.88)
LwF + R	49.70 (+5.49)	59.25 (+3.36)	56.45 (+4.22)	85.49 (+3.85)	66.82 (+1.02)	63.54 (+3.59)
D + R(**ours**)	**50.58 (+6.38)**	**60.52 (+4.63)**	**56.84 (+4.62)**	**86.00 (+4.36)**	**68.41 (+2.61)**	**64.47 (+4.52)**

	Imagenet → Birds → Flowers → Aircrafts → Scenes					
	Imagenet	Birds	Flowers	Aircrafts	Scenes	Average
LwF [15]	43.37 (ref)	52.26 (ref)	79.91 (ref)	63.25 (ref)	60.82 (ref)	59.92 (ref)
D (**ours**)	45.94 (+2.57)	51.90 (-0.36)	81.21 (+1.30)	64.30 (+1.05)	60.90 (+0.07)	60.85 (+0.93)
LwF + R	50.05 (+6.67)	55.60 (+3.34)	85.12 (+5.20)	66.43 (+3.18)	62.39 (+1.57)	63.92 (+3.99)
D + R(**ours**)	**50.84 (+7.47)**	**57.05 (+4.79)**	**85.72 (+5.81)**	**67.42 (+4.17)**	**62.91 (+2.09)**	**64.79 (+4.87)**

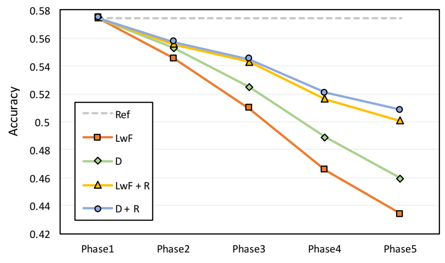
	Imagenet → Flowers → Aircrafts → Scenes → Birds					
	Imagenet	Flowers	Aircrafts	Scenes	Birds	Average
LwF [15]	44.49 (ref)	77.50 (ref)	61.57 (ref)	60.30 (ref)	56.02 (ref)	59.98 (ref)
D (**ours**)	46.37 (+1.88)	79.25 (+1.74)	62.47 (+0.90)	60.00 (-0.30)	57.22 (+1.21)	61.06 (+1.08)
LwF + R	50.26 (+5.77)	84.48 (+6.98)	65.38 (+3.81)	62.31 (+2.01)	57.54 (+1.52)	63.99 (+4.02)
D + R(**ours**)	**50.76 (+6.26)**	**85.07 (+7.56)**	**65.83 (+4.26)**	**62.54 (+2.24)**	**59.52 (+3.50)**	**64.74 (+4.76)**

From the results in Table 2 and Table 3, we observe that, either with or without *Retrospection*, *Distillation* outperforms *LwF* on each individual task. It demonstrates that adapting to the new task by knowledge distillation can facilitate the learning on the new task and simultaneously benefit the performance preservation on the old task. In some cases, *e.g.*, ImageNet→Birds, the performance of *Distillation* on the new task is superior to the reference provided by *Finetuning.* As far as we can observe, one reason is due to the regularization caused by mimicking the output of *Original CNN* as suggested in [15]. Another reason is that soft labels for the new task can not only enforce the relationship among classes [24] but also reduce the overfitting on new data, thus making the resulting model generalize better.

Besides, we respectively evaluate the effect of *Retrospection* on *LwF* and *Distillation.* The results indicate that a small subset of old data is beneficial for the performance preservation on the old task and also helpful for the learning on the new task.



(a) Imagenet→Scenes→Birds→Flowers→Aircrafts.



(b) Imagenet→Birds→Flowers→Aircrafts→Scenes.

Fig. 3. Accuracy degradation on ImageNet in five-task scenario. D for *Distillation*, and R for *Retrospection*.

Five-task Scenario. Table 4 displays the final accuracy by different methods in five-task scenario. *LwF* is treated as a strong baseline here. It can be seen that *Distillation* also works reasonably well with a longer sequence of tasks, and achieves superior (or at least comparable) performance to *LwF* on each individual task either with or without *Retrospection*. For a thorough comparison, we also illustrate the degradation of accuracy on ImageNet as the number of tasks grows in Figure 3, where the curve of *Distillation+Retrospection* goes down in the slowest rate. *Retrospection* further demonstrates its effectiveness for the performance preservation on old tasks. It is noteworthy that, with the help of *Retrospection*, on ImageNet which comes as the first task, the final accuracy with our method outperforms that with *LwF* by more than 6% in all three cases of five-task scenario shown in Table 4.

Comparison with *Encoder-based-LwF*. In Table 5, we compare *Distillation* to an orthogonal method denoted by *Encoder-based-LwF* [20]. It builds on the top of *LwF* and adds an auto-encoder for each old task, which aims at preserving the crucial features for old tasks at the cost of slightly increasing the model size.

We first carry on the experiments following the settings in [20], *i.e.*, the data for old tasks is not available[6]. The gain brought by *Encoder-based-LwF* compared to *LwF* is mainly for the old tasks and the performance on the latest task is comparable or a little inferior. In the sequence of five-task scenario shown in Table 5, *Distillation* is inferior to *Encoder-based-LwF* in the first two tasks but superior in the last three tasks, resulting in the comparable average performance. Moreover, the auto-encoder introduced by [20] can also be integrated with *Distillation*, which can further improve the accuracy.

[6] The results with *Encoder-based-LwF* in Table 5 are from our re-implementation, which basically agree with those in [20]. The models in [20] are implemented with MatConvnet [25] and the data augmentation is adopted when recording the output of *Original CNN*. Besides the case of five-task scenario, we also take the experiments in the two-task scenario, which are provided in the supplementary material.

Table 5. Classification accuracy (%) for comparison with *Encoder-based-LwF*. The reference performances are respectively given by *LwF* and *LwF+R*. *D* for *Distillation*, *R* for *Retrospection*, and *Encoder* for the approach in [20].

	Imagenet → Scenes → Birds → Flowers → Aircrafts					
	Imagenet	Scenes	Birds	Flowers	Aircrafts	Average
LwF [15]	44.20 (ref)	55.90 (ref)	52.22 (ref)	81.64 (ref)	65.80 (ref)	59.95 (ref)
LwF + Encoder [20]	46.35 (+2.14)	58.43 (+2.54)	52.95 (+0.72)	82.03 (+0.39)	64.75 (-1.05)	60.90 (+0.95)
D (**ours**)	46.15 (+1.95)	55.67 (-0.22)	53.17 (+0.95)	82.37 (+0.73)	66.79 (+0.99)	60.83 (+0.88)
D + Encoder (**ours**)	47.61 (+3.40)	57.76 (+1.86)	53.71 (+1.48)	82.56 (+0.93)	66.43 (+0.63)	**61.61 (+1.66)**

	Imagenet → Scenes → Birds → Flowers → Aircrafts					
	Imagenet	Scenes	Birds	Flowers	Aircrafts	Average
LwF+R	49.70 (ref)	59.25 (ref)	56.45 (ref)	85.49 (ref)	66.82 (ref)	63.54 (ref)
LwF+Encoder+R	50.47 (+0.77)	60.00 (+0.75)	56.45 (ref)	85.23 (-0.26)	66.46 (-0.36)	63.72 (+0.18)
D+R (**ours**)	50.58 (+0.89)	60.52 (+1.27)	56.84 (+0.40)	86.00 (+0.50)	68.41 (+1.59)	64.47 (+0.93)
D+Encoder+R (**ours**)	51.21 (+1.51)	61.49 (+2.24)	57.22 (+0.78)	86.04 (+0.55)	68.20 (+1.38)	**64.83 (+1.29)**

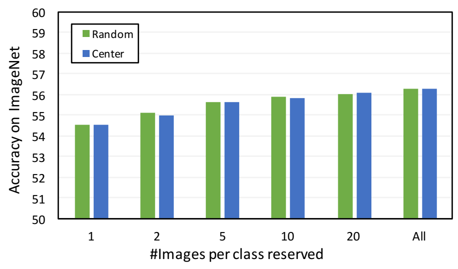
Then the experiments are further conducted with *Retrospection*, *i.e.*, a small subset of data is reserved for old tasks. The results are shown in the bottom half of Table 5. With an auto-encoder [20] incorporated for each old task, *Retrospection* is still much useful for performance preservation compared to those without revisiting the data for old tasks. Besides, the combination of *Distillation+Retrospection* and the auto-encoder [20] leads to the best result.

4.3 Discussion

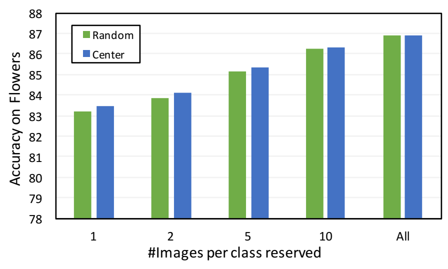
Retrospection Strategy. In our experimental settings for *Retrospection*, we randomly select five images per class for old tasks, and the results with different random seeds are consistent. Here we further conduct an ablation study to investigate the number of images per class reserved for old tasks and the sampling strategy. Imagenet→Birds and Flowers→Birds are taken as the benchmarks. The results are obtained by *Distillation+Retrospection*. The performance on the old task is adopted as the criterion here.

As shown in Figure 4, the performance on the old task rises as the number of stored images per class increases. Conserving five images per class for old tasks is a reasonable trade-off between performance and memory consumption. As for the sampling strategy, in addition to random selection, here we attempt another sampling strategy. Specifically, a class center is first computed for each class by averaging the features of all samples belonging to this class, and then the images close to the class center are selected for *Retrospection*. The results in Figure 4 indicate that this strategy does not show significant superiority to random selection. It is worth further exploration to develop more effective strategies for *Retrospection*, *e.g.*, to discover the number of images for each class adaptively.

Computation Cost. The computation cost introduced by *Distillation* compared to *LwF* [15] lies in two aspects: training *Expert CNN* on the new task and then recording its output, neither of which is cumbersome. The target model



(a) Imagenet→Birds



(b) Flowers→Birds

Fig. 4. Ablation study on *Retrospection* strategy. Random for random selection, and Center for selecting images close to the class center. The accuracy on the old task increases with the increasing number of images reserved for each class. Choosing images close to the class center is not significantly superior to random selection.

size is not increased at all. As for *Retrospection*, it requires additional memory space to store the data of old tasks. Nevertheless, our study shows that a small subset of old data can greatly benefit the performance preservation on old tasks, especially in a long sequence of tasks for different domains. For example, in the first case of five-task scenario shown in Table 4, the top-1 accuracy on ImageNet with *LwF+Retrospection* outperforms that with *LwF* by 5.49%, while the 5000 images reserved for ImageNet is less than 1/240 of the total training set.

5 Conclusion

This work proposes a novel approach, consisting of *Distillation* and *Retrospection*, for multi-task lifelong learning, which strikes a better balance on the performance preservation on old tasks and the adaptation to a new task. *Adaptation by Distillation* from an intermediate *Expert CNN* can not only facilitate the learning on the new task but also is beneficial for preserving the performance on old tasks. *Retrospection* is proposed to cache a small subset of data for old tasks, which proves to be greatly helpful for the performance preservation, especially in long sequences of tasks drawn from different distributions. The combination of *Distillation* and *Retrospection* outperforms *LwF* by a large margin, and bring consistent improvements to both old and new tasks.

Acknowledgment: This work is partially supported by the NSFC under Grant 61673362, Youth Innovation Promotion Association CAS, and the Fundamental Research Funds for the Central Universities. This work is also partially supported by the Big Data Collaboration Research grant from SenseTime Group (CUHK Agreement No. TS1610626), the General Research Fund (GRF) of Hong Kong (No. 14236516, 14241716, 14224316, 14209217).

References

1. http://nbviewer.jupyter.org/github/BVLC/caffe/blob/master/examples/02-fine-tuning.ipynb
2. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. arXiv preprint arXiv:1711.09601 (2017)
3. Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: Lifelong learning with a network of experts. In: CVPR (2017)
4. Caruana, R.: Multitask learning. In: Learning to learn, pp. 95–133. Springer (1998)
5. Chen, T., Goodfellow, I., Shlens, J.: Net2net: Accelerating learning via knowledge transfer. In: ICLR (2016)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
7. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: ICML (2014)
8. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
9. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211 (2013)
10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
11. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
12. Jung, H., Ju, J., Jung, M., Kim, J.: Less-forgetful learning for domain expansion in deep neural networks. In: AAAI (2018)
13. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences **114**(13), 3521–3526 (2017)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
15. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
16. Maji, S., Kannala, J., Rahtu, E., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. Tech. rep. (2013)
17. Mitchell, T.M.: The need for biases in learning generalizations. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ. New Jersey (1980)
18. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (2008)
19. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: CVPR (2009)
20. Rannen Ep Triki, A., Aljundi, R., Blaschko, M., Tuytelaars, T.: Encoder based lifelong learning. In: ICCV (2017)
21. Rebuffi, S.A., Kolesnikov, A., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: CVPR (2017)

22. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. In: ICLR (2015)
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
24. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: ICCV (2015)
25. Vedaldi, A., Lenc, K.: Matconvnet – convolutional neural networks for matlab. In: ACM Multimedia (2015)
26. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
27. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: ICML (2017)