Incremental Learning In Online Scenario

Jiangpeng He
he416@purdue.edu

Runyu Mao
mao111@purdue.edu

Zeman Shao
shao112@purdue.edu

Fengqing Zhu
zhu0@purdue.edu

School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana USA

Abstract

Modern deep learning approaches have achieved great success in many vision applications by training a model using all available task-specific data. However, there are two major obstacles making it challenging to implement for real life applications: (1) Learning new classes makes the trained model quickly forget old classes knowledge, which is referred to as catastrophic forgetting. (2) As new observations of old classes come sequentially over time, the distribution may change in unforeseen way, making the performance degrade dramatically on future data, which is referred to as concept drift. Current state-of-the-art incremental learning methods require a long time to train the model whenever new classes are added and none of them takes into consideration the new observations of old classes. In this paper, we propose an ***incremental learning framework*** *that can work in the challenging online learning scenario and handle both new classes data and new observations of old classes. We address problem (1) in online mode by introducing a modified cross-distillation loss together with a two-step learning technique. Our method outperforms the results obtained from current state-of-the-art offline incremental learning methods on the CIFAR-100 and ImageNet-1000 (ILSVRC 2012) datasets under the same experiment protocol but in online scenario. We also provide a simple yet effective method to mitigate problem (2) by updating exemplar set using the feature of each new observation of old classes and demonstrate a real life application of online food image classification based on our complete framework using the Food-101 dataset.*

1. Introduction

One of the major challenges of current deep learning based methods when applied to real life applications is learning new classes incrementally, where new classes are continuously added overtime. Furthermore, in most real life scenarios, new data comes in sequentially, which may contain both the data from new classes or new observations of old classes. Therefore, a practical vision system is expected to handle the data streams containing both new and old classes, and to process data sequentially in an online learning mode [15], which has similar constrains as in real life applications. For example, a food image recognition system designed to automate dietary assessment should be able to update using each new food image continually without forgetting the food categories already learned.

Most deep learning approaches trained on static datasets suffer from the following issues. First is catastrophic forgetting [16], a phenomenon where the performance on the old classes degrades dramatically as new classes are added due to the unavailability of the complete previous data. This problem become more severe in online scenario due to limited run-time and data allowed to update the model. The second issue arises in real life application where the data distribution of already learned classes may change in unforeseen ways [23], which is related to concept drift [5]. In this work, we aim to develop an incremental learning framework that can be deployed in a variety of image classification problems and work in the challenging online learning scenario.

A practical deep learning method for classification is characterized by (1) its ability to be trained using data streams including both new classes data and new observations of old classes, (2) good performance for both new and old classes on future data streams, (3) short run-time to update with constrained resources, and (4) capable of lifelong learning to handle multiple classes in an incremental fashion. Although progress has been made towards reaching these goals [14, 21, 2, 31], none of the existing approaches for incremental learning satisfy all the above conditions. They assume the distribution of old classes data remain unchanged overtime and consider only new classes data for incoming data streams. As we mentioned earlier, data distribution are likely to change in real life[23]. When concept drift happens, regardless the effort put into retaining the old classes knowledge, degradation in performance is inevitable. In addition, although these existing methods have achieved state-of-the-art results, none of them work in the challenging online scenario. They require offline training

2575-7075/20/$31.00 ©2020 IEEE
DOI 10.1109/CVPR42600.2020.01394

using all available new data for many epochs, making it impractical for real life applications.

The main contributions of this paper is summarized as follows.

- We introduce a modified cross-distillation loss together with a two-step learning technique to make incremental learning feasible in online scenario. We show comparable results to the current state-of-the-art [21, 2, 31] on CIFAR-100 [12] and ImageNet-1000 (ILVSC2012) [25]. We follow the same experiment benchmark protocol [21] where all new data belong to new class, but in the challenging online learning scenario where the condition is more constrained for both run-time and number of data allowed to update the model.

- We propose an incremental learning framework that is capable of lifelong learning and can be applied to a variety of real life online image classification problems. In this case, we consider new data belong to both new class and existing class. We provide a simple yet effective method to mitigate concept drift by updating the exemplar set using the feature of each new observation of old classes. Finally, we demonstrate how our complete framework can be implemented for food image classification using the Food-101 [1] dataset.

2. Related Work

In this section, we review methods that are closely related to our work. Incremental learning remains one of the long-standing challenges for machine learning, yet it is very important to brain-like intelligence capable of continuously learning and knowledge accumulation through its lifetime.

Traditional methods. Prior to deep learning, SVM classifier [4] is commonly used. One representative work is [24], which learns the new decision boundary by using support vectors that are learned from old data together with new data. An alternative method is proposed in [3] by retaining the Karush-Kuhn-Tucker conditions instead of support vectors on old data and then update the solution using new data. Other techniques [19, 17, 13] use ensemble of weak classifiers and nearest neighbor classifier.

Deep learning based methods. These methods provide a joint learning of task-specific features and classifiers. Approaches such as [10, 11] are based on constraining or freezing the weights in order to retain the old tasks performance. In [10], the last fully connected layer is freezed which discourages change of shared parameters in the feature extraction layers. Inn [11] old tasks knowledge is retained by constraining the weights that are related to these tasks. However, constraining or freezing parameters also limits its adaptability to learn from new data. A combination of knowledge distillation loss [9] with standard cross-entropy loss is proposed to retain the old classes knowledge in [14], where old and new classes are separated in multi-class learning and distillation is used to retain old classes performance. However, performance is far from satisfactory when new classes are continuously added, particularly in the case when the new and old classes are closely related. Based on [14], auto encoder is used to retain the knowledge for old classes instead of using distillation loss in [20]. For all these methods, only new data is considered.

In [26] and [28], synthetic data is used to retain the knowledge for old classes by applying a deep generative model [6]. However, the performance of these methods are highly dependent on the reliability of the generative model, which struggles in more complex scenarios.

Rebuffi et al proposed iCaRL[21], an approach using a small number of exemplars from each old class to retain knowledge. An end-to-end incremental learning framework is proposed in [2] using exemplar set as well, along with data augmentation and balanced fine-tuning to alleviate the imbalance between the old and new classes. Incremental learning for large datasets was proposed in [31] in which a linear model is used to correct bias towards new classes in the fully connected layer. However, it is difficult to apply these methods to real life applications since they all require a long offline training time with many epochs at each incremental step to achieve a good performance. In addition, they assume the distribution of old classes remain unchanged and only update the classifiers using new classes data. All in all, a modified cross-distillation loss along with a two-step learning technique is introduced to make incremental learning feasible in the challenging online learning scenario. Furthermore, our complete framework is capable of lifelong learning from scratch in online mode, which is illustrated in Section 4.

3. Online Incremental Learning

Online incremental learning [15] is a subarea of incremental learning that are additionally bounded by run-time and capability of lifelong learning with limited data compared to offline learning. However, these constraints are very much related to real life applications where new data comes in sequentially and is in conflict with the traditional assumption that complete data is available. A sequence of model $h_1, h_2, ..., h_t$ is generated on the given stream of data blocks $s_1, s_2, ..., s_t$ as shown in Figure 1. In this case, s_i is a block of new data with block size p, defined as the number of data used to update the model, which is similar to batch size as in offline learning mode. However, each new data is used only once to update the model instead of training the model using the new data with multiple epochs as in offline mode. $s_t = \{(\mathbf{x}_t^{(1)}, y_t^{(1)}), ..., (\mathbf{x}_t^{(p)}, y_t^{(p)})\} \in R^n \times \{1, ..., M\}$ where n is the data dimension and M is the total number of classes. The model $h_t : R^n \to \{1, ..., M\}$

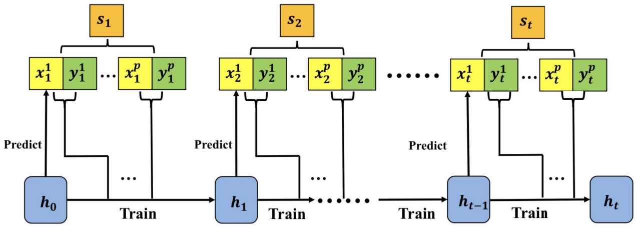

s_1
s_2
s_t
x_1^1 y_1^1 … x_1^p y_1^p
x_2^1 y_2^1 … x_2^p y_2^p
x_t^1 y_t^1 … x_t^p y_t^p
Predict
Predict
Predict
h_0
h_1
h_{t-1}
h_t
Train
Train
Train
Train

Figure 1: **Online Scenario.** A sequence of model $h_1, h_2, ..., h_t$ is generated using each block of new data with block size p, where $(\mathbf{x}_t^i, y_t^i)$ indicate the i-th new data for the t-th block.

depends solely on the model h_{t-1} and the most recent block of new data s_t consisting of p examples with p being strictly limited, e.g. if we set $p = 16$ then we will predict for each new data and use a block of 16 new data to update the model.

Catastrophic forgetting is the main challenge faced by all incremental learning algorithms. Suppose a model h_{base} is initially trained on n classes and we update it with m new added classes to form the model h_{new}. Ideally, we hope h_{new} can predict all $n + m$ classes well, but in practice the performance on the n old classes drop dramatically due to the lack of old classes data when training the new classes. In this work, we propose a modified cross-distillation loss and a two-step learning technique to address this problem in online scenario.

Concept drift is another problem that happens in most real life applications. Concept [29] in classification problems is defined as the joint distribution $P(X, Y)$ where X is the input data and Y represents target variable. Suppose a model is trained on data streams by time t with joint distribution $P(X_t, Y_t)$, and let $P(X_n, Y_n)$ represent the joint distribution of old classes in future data streams. Concept drift happens when $P(X_t, Y_t) \neq P(X_n, Y_n)$. In this work, we do not measure concept drift quantitatively, but we provide a simple yet effective method to mitigate the problem by updating the exemplar set using the features of each new data in old classes, which is illustrated in Section 4.3

4. Incremental Learning Framework

In this work, we propose an incremental learning framework as shown in Figure 2 that can be applied to any online scenario where data is available sequentially and the network is capable of lifelong learning. There are three parts in our framework: *learn from scratch*, *offline retraining* and *learn from a trained model*. Incremental learning in online scenario is implemented in 4.3 and lifelong learning can be achieved by alternating the last two parts after initial learning.

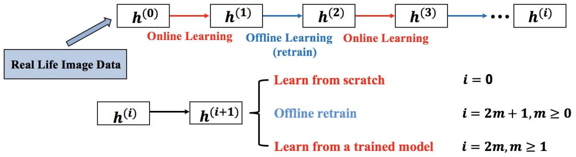


Figure 2: **Proposed incremental learning framework.** $h^{(i)}$ indicates the evolving model at i-th step.

4.1. Learn from Scratch

This part serves as the starting point to learn new classes. In this case, we assume the network does not have any previous knowledge of incoming classes, which means there is no previous knowledge to be retained. Our goal is to build a model that can adapt to new classes fast with limited data, e.g. block size of 8 or 16.

Baseline. Suppose we have data streams with block size p belong to M classes: $\{s_1, ..., s_t\} \in R^n \times \{1, ..., M\}$. The baseline for the model to learn from sequential data can be thought as generating a sequence of model $\{h_1, ..., h_t\}$ using standard cross-entropy where h_t is updated from h_{t-1} by using block of new data s_t. Thus h_t is evolving from h_0 for a total of t updates by using the given data streams. Compared to traditional offline learning, the complete data is not available and we need to update the model for each block of new data to make it dynamically fit to the data distribution used so far. So in the beginning, the performance on incoming data is poor due to data scarcity.

Online representation learning. A practical solution is to utilize representation learning when data is scarce at the beginning of the learning process. Nearest class Mean (NCM) classifier [22, 21] is a good choice where the test image is classified as the class with the closest class data mean. We use a pre-trained deep network to extract features by adding a representation layer before the last fully connected layer for each input data $\mathbf{x}_i$ denoted as $\phi(\mathbf{x}_i)$. Thus the classifier can be expressed as

$$y^* = \underset{y \in \{1,...,M\}}{\arg\min} \quad d(\phi(\mathbf{x}), \mu_y^\phi). \tag{1}$$

The class mean $\mu_y^\phi = \frac{1}{N_y} \sum_{i:y_i=i} \phi(\mathbf{x}_i)$ and N_y denote the number of data in classes y. We assume that the highly nonlinear nature of deep representations eliminates the need of a linear metric and allows to use Euclidean distance here

$$d_{xy}^\phi = (\phi(\mathbf{x}) - \mu_y^\phi)^T (\phi(\mathbf{x}) - \mu_y^\phi) \tag{2}$$

Our method: combining baseline with NCM classifier. NCM classifier behaves well when number of available data is limited since the class representation is based solely on the mean representation of the images belonging to that class. We apply NCM in the beginning and update using an online estimate of the class mean [7] for each new observation.

$$\mu_y^\phi \leftarrow \frac{n_{yi}}{n_{yi}+1}\mu_y^\phi + \frac{1}{n_{yi}+1}\phi(\mathbf{x}_i) \tag{3}$$

We use a simple strategy to switch from NCM to baseline classifier when accuracy for baseline surpass representation learning for s consecutive blocks of new data. Based on our empirical results, we set $s = 5$ in this work.

4.2. Offline Retraining

In order to achieve lifelong learning, we include an offline retraining part after each online incremental learning phase. By adding new classes or new data of existing class, both catastrophic forgetting and concept drift [5] become more severe. The simplest solution is to include a periodic offline retraining by using all available data up to this time instance.

Construct exemplar set. We use herding selection [30] to generate a sorted list of samples of one class based on the distance to the mean of that class. We then construct the exemplar set by using the first q samples in each class $\{E_1^{(y)}, ...E_q^{(y)}\}, y \in [1, ..., n]$ where q is manually specified. The exemplar set is commonly used to help retain the old classes' knowledge in incremental learning methods.

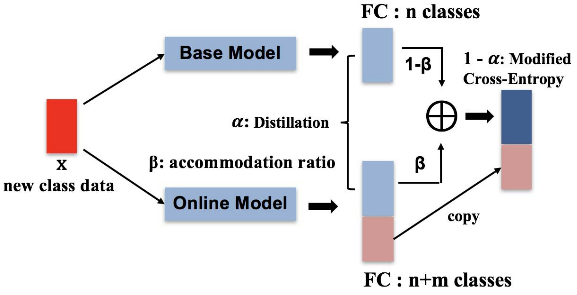


Figure 3: **Modified Cross-Distillation Loss.** It contains two losses: the distilling loss on old classes and the modified cross-entropy loss on all old and new classes.

4.3. Learn from a Trained Model

This is the last component of our proposed incremental learning framework. The goal here is to continue to learn from new data streams starting from a trained model. Different from existing incremental learning, we define new data containing both new classes data and new observations of old classes and we use each new data only once for training in online scenario. In additional to addressing the catastrophic forgetting problem, we also need to consider concept drift for already learned classes due to the fact that data distribution in real life application may change over time in unforeseen ways [23].

Baseline: original cross-distillation loss. Cross-distillation loss function is commonly used in state-of-the-art incremental learning methods to retain the previous learned knowledge. In this case, we consider only new classes data for incoming data streams. Suppose the model is already trained on n classes, and there are m new classes added. Let $\{(\mathbf{x}_i, y_i), y_i \in [n+1, ...n+m]\}$ denote new classes data. The output logits of the new classifier is denoted as $p^{(n+m)}(x) = (o^{(1)}, ..., o^{(n)}, o^{(n+1)}, ...o^{(n+m)})$, the recorded old classes classifier output logits is $\hat{p}^{(n)}(x) = (\hat{o}^{(1)}, ..., \hat{o}^{(n)})$. The knowledge distillation loss [9] can be formulated as in Equation 4, where $\hat{p}_T^{(i)}$ and $p_T^{(i)}$ are the i-th distilled output logit as defined in Equation 5

$$L_D(x) = \sum_{i=1}^{n} -\hat{p}_T^{(i)}(x) log[p_T^{(i)}(x)] \tag{4}$$

$$\hat{p}_T^{(i)} = \frac{\exp{(\hat{o}^{(i)}/T)}}{\sum_{j=1}^{n}\exp{(\hat{o}^{(j)}/T)}}, \; p_T^{(i)} = \frac{\exp{(o^{(i)}/T)}}{\sum_{j=1}^{n}\exp{(o^{(j)}/T)}} \tag{5}$$

T is the temperature scalar. When $T = 1$, the class with the highest score has the most influence. When $T > 1$, the remaining classes have a stronger influence, which forces the network to learn more fine grained knowledge from them. The cross entropy loss to learn new classes can be expressed as $L_C(x) = \sum_{i=1}^{n+m} -\hat{y}^{(i)} log[p^{(i)}(x)]$ where $\hat{y}$ is the one-hot label for input data x. The overall cross-distillation loss function is formed as in Equation 6 by using a hyper-parameter α to tune the influence between two components.

$$L_{CD}(x) = \alpha L_D(x) + (1-\alpha)L_C(x) \tag{6}$$

Modified cross-distillation with accommodation ratio. Although cross-distillation loss forces the network to learn latent information from the distilled output logits, its ability to retain previous knowledge still remains limited. An intuitive way to make the network retain previous knowledge is to keep the output from the old classes' classifier as a part of the final classifier. Let output logits of the new classifier be denoted as $p^{(n+m)}(x) = (o^{(1)}, ..., o^{(n)}, o^{(n+1)}, ...o^{(n+m)})$, the recorded old classes' classifier output logits is $\hat{p}^{(n)}(x) = (\hat{o}^{(1)}, ..., \hat{o}^{(n)})$. We use an accommodation ratio $0 \leq \beta \leq 1$ to combine the two classifier output as

$$\tilde{p}^{(i)} = \begin{cases} \beta p^{(i)} + (1-\beta)\hat{p}^{(i)} & 0 < i \leq n \\ p^{(i)} & n < i \leq n+m \end{cases} \tag{7}$$

When $\beta = 1$, the final output is the same as the new classifier and when $\beta = 0$, we replace the first n output units with the old classes classifier output. This can be thought as using the accommodation ratio β to tune the output units for old classes. As shown in Figure 3, the modified cross-distillation loss can be expressed by replacing the original cross-entropy loss part $L_C(x)$ with the new modified cross-entropy loss $\tilde{L}_C(x) = \sum_{i=1}^{n+m} -\hat{y}^{(i)} log[\tilde{p}^{(i)}(x)]$ after applying the accommodation ratio as in Equation 8

$$\tilde{L}_{CD}(x) = \alpha L_D(x) + (1-\alpha)\tilde{L}_C(x) \tag{8}$$

Algorithm 1 Update Exemplar Set

```
Input: New observation for old classes $(\mathbf{x}_i, y_i)$
Require: Old classes feature extractor $\Theta$
Require: Current exemplar set $\{E_1^{(y_i)}, ...E_q^{(y_i)}\}$
$M^{(y_i)} \leftarrow \frac{n_{y_i}}{n_{y_i}+1} M^{(y_i)} + \frac{1}{n_{y_i}+1} \Theta(\mathbf{x}_i)$
for m = 1,...,q do
    $d^{(m)} = (\Theta(E_m^{(y_i)}) - M^{(y_i)})^T (\Theta(E_m^{(y_i)}) - M^{(y_i)})$
$d_{min} \leftarrow \min\{d^{(1)}, ..., d^{(m)}\}$
$I_{min} \leftarrow \text{Index}\{d_{min}\}$
$d^{(q+1)} = (\Theta(\mathbf{x}_i) - M^{(y_i)})^T (\Theta(\mathbf{x}_i) - M^{(y_i)})$
if $d^{(q+1)} \leq d_{min}$ then
    Remove $E_{I_{min}}^{(y_i)}$ from $\{E_1^{(y_i)}, ...E_q^{(y_i)}\}$
    Add $x_i$ to $\{E_1^{(y_i)}, ...E_{q-1}^{(y_i)}\}$
else
    No need to update current exemplars
return $\{E_1^{(y_i)}, ...E_q^{(y_i)}\}$
```

We empirically set $\beta = 0.5$, $T = 2$ and $\alpha = \frac{n}{n+m}$ in this work where n and m are the number of old and new classes. The modified cross-distillation loss push the network to learn more from old classes' output units since we add it directly as part of the final output.

Update exemplar set. As described in Section 1, we consider the new data streams containing both new classes data and new observations of old classes with unknown distribution. In this case, retaining previous knowledge is not sufficient since concept drift can happen to old classes and the model will still undergo performance degradation. One solution is to keep updating the network using the exemplars for old classes. The class mean of each old class $\{M^{(1)}, ..., M^{(n)},\ M^{(i)} \in R^n\}$ is calculated and recorded as described in Section 4.2 by constructing the exemplar set $\{(E_1^{(y)}, ...E_q^{(y)}), y \in [1, ..., n]\}$ using previous data streams. Let $\{(\mathbf{x}_i, y_i), y_i \in [1, ..., n]\}$ denote the new observation of old classes. We follow the same online class mean update as described in Equation 3 to keep updating the class mean with all data seen so far. So when concept drift happens, e.g., the class mean changes toward the new data, we update the exemplar set to make new data more likely to be selected to update the model during two-step learning as described in next part. The complete process of updating exemplar set is shown in Algorithm 1.

Two-step learning. Unlike other incremental learning algorithms that mix new classes data with old classes exemplars, we first let the model learn from a block of new classes data and then a balanced learning step is followed. This two-step learning technique is deigned for online learning scenarios, where both update time and number of available data are limited. As shown in Figure 5, we use the modified cross-distillation loss in the first step to overcome catastrophic forgetting since all data in this block belongs to new classes. In the second step, we pair same number of old classes exemplars from the exemplar set with the new classes data. As we have balanced new and old classes, cross entropy loss is used to achieve balanced learning.

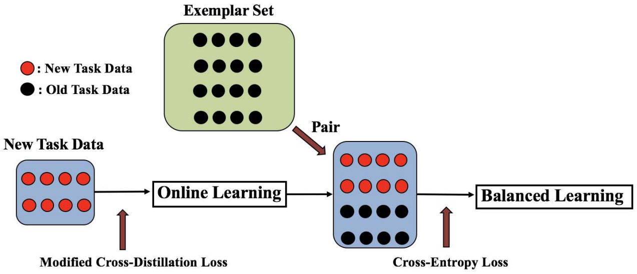


Figure 5: **Two-Step Learning.** Black dots correspond to old classes samples stored in exemplar set. Red dots correspond to samples from new classes.

5. Experimental Results

Our experimental results consists of two main parts. In part one, we compare our modified cross-distillation loss and the two-step learning technique as introduced in Section 4.3 with current state-of-the-art incremental learning methods [2, 14, 31, 21]. We follow the iCaRL experiment benchmark protocol [21] to arrange classes and select exemplars, but in the more challenging online learning scenario as illustrated in Section 5.3. Our method is implemented on two public datasets: **CIFAR-100** [12] and **ImageNet-1000** (ILSVRC 2012) [25]. Part two is designed to test the performance of our complete framework. Since our goal is to set up an incremental learning framework that can be applied to online learning scenario, we use **Food-101** [1] food image dataset to evaluate our methods. For each part of our proposed framework, we compare our results to baseline methods as described in Section 4.

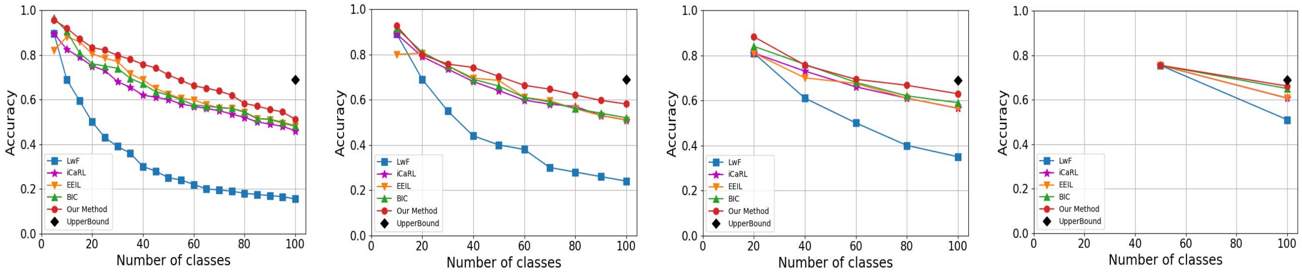
5.1. Datasets

We used three public datasets. Two common datasets: CIFAR-100 and ImageNet-1000 (ILSVRC 2012) and one food image dataset: Food-101.

Food-101 is the largest real-world food recognition dataset consisting of 1k images per food classes collected from *foodspotting.com*, comprising of 101 food classes. We divided 80% for training and 20% for testing for each class.

CIFAR-100 consists of 60k 32×32 RGB images for 100 common objects. The dataset is originally divided into 50K as training and 10k as testing.

ImageNet-1000 (ILSVRC 2012) ImageNet Large-Scale Visual Recognition Challenge 2012 (ILSVRC12) is an annual competition which uses a subset of ImageNet. This subset contains 1000 classes with more than 1k images per



(a) (b) (c) (d)

Figure 4: **Incremental learning results on CIFAR-100** with split of (a) 5 classes, (b) 10 classes, (c) 20 classes and (d) 50 classes. The **Upper Bound** in last step is obtained by offline training a model using all training samples from all classes. (Best viewed in color)

class. In total, there are about 1.2 million training data, 50k validation images, and 150k testing images.

Data pre-processing For Food-101, we performed image resize and center crop. As for CIFAR-100, random cropping and horizontal flip was applied following the original implementation [8]. For ImageNet, we follow the steps in VGG pre-processing [27], including random cropping, horizontal flip, image resize and mean subtraction.

5.2. Implementation Detail

Our implementation is based on Pytorch [18]. For experiment part one, we follow the same experiment setting as current state-of-the-art incremental learning methods, a standard 18-layer ResNet for ImageNet-1000 and a 32-layer ResNet for CIFAR-100. For experiment part two, we applied a 18-layer ResNet to Food-101. The ResNet implementation follows the setting suggested in [8]. We use stochastic gradient descent with learning rate of 0.1, weight decay of 0.0001 and momentum of 0.9.

Selection of block size p in online learning scenario. Different from offline learning scenario, where we select a batch size to maximize overall performance after many epochs. In online learning scenario, we need to select block size p based on real life applications. More specifically, a large block size causes slow update since we have to wait until enough data arrives to update the model. On the other hand, using a very small block size, e.g., update with each new observation, although very fast, is not suitable for deep neural network due to strong bias towards new data. Therefore, we design a pretest using the first 128 new data for each experiment to repeatedly update the model by varying block size $p \in \{1, 2, 4, 8, 16, 32, 64\}$. Thus the optimal block size is chosen which gives the highest accuracy on these 128 new data. We do not consider $p > 64$ as such a large block size is not practical for real life applications.

5.3. Evaluation of Modified Cross-Distillation Loss and Two-Step Learning

In this part, we compared our modified cross-distillation loss and two-step learning technique with the current state-of-the-art methods [21, 2, 31]. We consider the online setting that new classes data comes sequentially and we predict each new data at first and then use a block of new data to update the model. For each incremental step, we compare our accuracy obtained in online scenario with state-of-the-art results in offline mode. We constructed the exemplar set for both CIFAR and ImageNet with the same number of samples as in [21, 2, 31] for fair comparison.

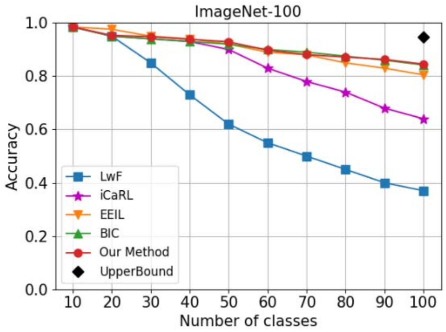
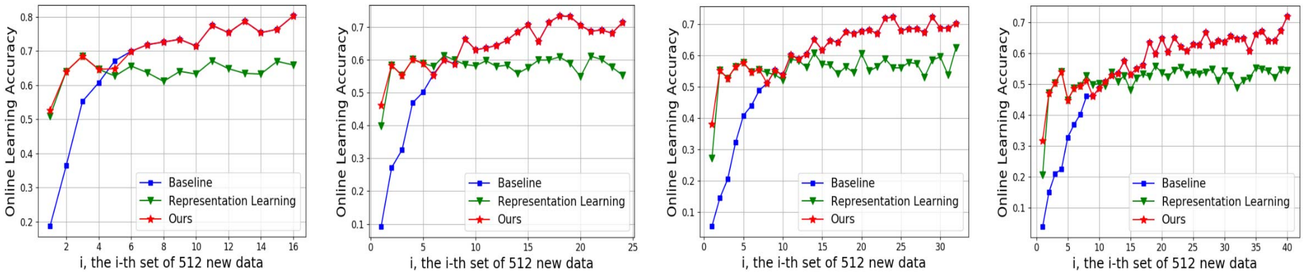


Figure 7: **Incremental learning results on ImageNet-100** with split of 10 classes. The **Upper Bound** in last step is obtained by offline training a model using all training samples from all classes. (Best viewed in color)

CIFAR-100. We divided 100 classes into splits of 5, 10, 20, and 50 in random order. Therefore, we have incremental training steps for 20, 10, 5, and 2, respectively. The optimal block size is set as $p = 8$. We ran the experiment for four trials and each time with a random order for the 100 classes. The average accuracy is shown in Figure 4. Our method shows the best accuracy for all incremental learning steps even in the challenging online learning scenario.

ImageNet-1000. As 1000-class is too large and impractical for online scenario, so we randomly selected 100 classes from the 1000 classes to construct a subset of the original dataset, which is referred to as ImageNet-100. We then divided the 100 classes into 10 classes split so we have an incremental step of 10. The optimal block size is set as $p = 16$. We ran this for four trials as before and we recorded the average accuracy in each step as shown in Figure 7. Although the performance of EEIL [2] surpass our



(a) (b) (c) (d)

Figure 6: **Starting from scratch on Food-101** with number of new classes (a) 20 classes (b) 30 classes (c) 40 classes and (d) 50 classes. Baseline and our method are illustrated in Section 4.1 (Best viewed in color)

Method	20	30	40	50
Baseline	62.81%	56.53%	54.35%	51.39%
Representation Learning	60.21%	55.32%	53.68%	51.26%
Ours	**70.90%**	**64.32%**	**62.31%**	**57.83%**

(a)

	Testing	Upper Bound
20	78.77%	84.17%
30	73.28%	80.95%
40	71.42%	77.82%
50	67.54%	74.46%

(b)

Table 1: **Online learning from scratch on Food-101** with (a) Online accuracy and (b) Testing accuracy. The **Upper Bound** is obtained by offline training a model using all training samples from all given classes. (Best result marked in bold)

method in the second step, we attain the best performance as more classes are added.

5.4. Evaluation of Our Complete Framework

We used a food image dataset **Food-101** [1] to evaluate performance of our proposed incremental learning framework.

Benchmark protocol of online incremental learning. Until now, there is no benchmark protocol on how to evaluate an online incremental learning method. In addition to address catastrophic forgetting [16] as in offline incremental learning, we also need to consider concept drift [5] in online scenario. We propose the following evaluation procedure: for a given multi-class classification dataset, the classes should be randomly arranged. For each class, the training data should be further split into new training data and old training data. The former is used when a class is introduced to the model for the first time. The later is considered when the model has seen the class before, which is used to simulate real life applications and test the ability of the method to handle new observations of old classes. After each online learning phase, the updated model is evaluated on test data containing all classes already been trained so far. There is no over-fitting since the test data is never used to update the model. In addition to the overall test accuracy, we should separately examine the accuracy for new classes and accuracy for old classes data. We also suggest to use online accuracy, which is the accuracy for data in training set before they are used to update the model, to represent the classification performance of future data stream. In general, online accuracy shows the model's ability to adapt to future data stream and online accuracy for old classes indicates the model's ability to handle new observations of old classes.

5.5. Results on Food-101

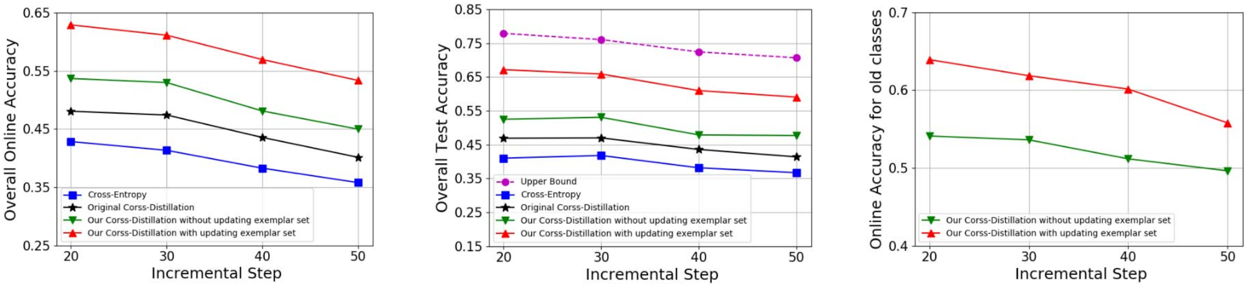
Although there are three separate components of the proposed incremental learning framework as described in Section 4, we only test the component described in 4.1 once and then alternate between the two components described in 4.2 and 4.3. In addition, the offline retraining part in 4.2 is inapplicable with online incremental learning. So in this experiment, we test for one cycle of our proposed framework starting from scratch then learning from a trained model provided by offline retraining. We use half training data per class as new classes data and the other half as new observations of old classes. We divided the Food-101 dataset into split of 20, 30, 40, 50 classes randomly and performed the one incremental step learning with step size of 20, 30, 40, and 50, respectively. In addition, we construct exemplar set with only 10 samples per class to simulate real life applications instead of including more samples per class.

Learn from scratch. In this part, we evaluate our method that combines baseline and representation learning as described in Section 4.1. Optimal block size is set as $p = 16$. Result of online accuracy compared to baseline and representation learning is shown in Table 1a. Our method achieved the best online accuracy in all incremental learning steps. Similarly, test accuracy compared to upper bound is shown in Table 1b. We also calculated the accuracy of each 512 incoming new data as shown in Figure 6. We observed that the representation learning works well at the beginning when data is scarce and the baseline achieved higher accuracy as the number of new data increases. Thus by combining the two methods and automatically switch from one to the other, we attain a higher overall online accuracy.

Learn from a trained model. In this part, we perform

	Online Accuracy		Test Accuracy	
Incremental Step	new	old	new	old
20	54.35% → **64.78%**	22.83% → **61.01%**	**70.97%** → 64.00%	41.77% → **70.32%** (84.17%)
30	52.62% → **62.25%**	22.41% → **60.00%**	**71.56%** → 61.87%	42.25% → **69.90%** (80.95%)
40	46.30% → **61.53%**	20.53% → **53.43%**	**66.62%** → 56.31%	40.82% → **65.65%** (77.82%)
50	43.49% → **56.76%**	19.47% → **51.71%**	**63.32%** → 54.20%	36.81% → **63.92%** (74.46%)

Table 2: **Online learning from a trained model on Food-101** with **baseline method using original cross-distillation loss** in the left of → and **our proposed method** in the right (best result marked in bold), (·) shows the **Upper Bound** results.



(a) (b) (c)

Figure 8: **Ablation study on Food-101 dataset** (a) overall online accuracy (b) overall test accuracy (c) online accuracy for old classes. (Best viewed in color)

a one incremental step experiment following our proposed benchmark protocol described in Section 5.4 and the result is shown in Table 2. Compared to the baseline, our method improved the online learning accuracy for both new and old classes, which shows that our model can adapt quickly to future data stream including both new classes data or new observations of old classes. In addition, we significantly improved the test accuracy compared to the baseline method. However, the trade off is slightly lower accuracy for the new classes test accuracy compared to the baseline due to the use of the accommodation ratio in our method. Since it is difficult for the model to perform well on new classes without losing knowledge from the old classes, the accommodation ratio can be manually tuned to balance between the new classes and the old classes depending on the application scenario. A higher accommodation ratio leads to higher accuracy on new classes by trading off accuracy on old classes. For this experiment, we simply use $\beta = 0.5$.

Ablation study. We analyzed different components of our method to demonstrate their impacts. We first show the influence of different loss functions including cross-entropy, cross-distillation, and our modified cross-distillation. We then analyzed the impact of updating the exemplar set to mitigate concept drift. As shown in Figure 8a and 8b, even without updating exemplar set, our modified cross-distillation loss outperformed the other two (black and blue lines) for all incremental steps. By updating the exemplar set, we were able to achieve a higher overall online and test accuracy. Furthermore, Figure 8c illustrates improvement of online accuracy for old classes by updating the exemplar set. Since we do not deliberately select any new data from old classes to update the model during the incremental learning step, as the data distribution changes, our method was able to automatically update the exemplar set by using the current class mean calculated by all data in old classes seen so far. Thus through the proposed two-step learning which pairs each new data with an exemplar, we can achieve a higher online accuracy for future data streams.

6. Conclusion

In this paper, we proposed an incremental learning framework including a modified cross-distillation loss together with a two-step learning technique to address catastrophic forgetting in the challenging online learning scenario, and a simple yet effective method to update the exemplar set using the feature of each new observation of old classes data to mitigate concept drift. Our method has the following properties: (1) can be trained using data streams including both new classes data and new observations of old classes in online scenario, (2) has good performance for both new and old classes on future data streams, (3) requires short run-time to update with limited data, (4) has potential to be used in lifelong learning that can handle unknown number of classes incrementally. Our method outperforms current state-of-the-art on CIFAR-100 and ImageNet-1000 (ILSVRC 2012) in the challenging online learning scenario. Finally, we showed our proposed framework can be applied to real life image classification problem by using Food-101 dataset as an example and observed significant improvement compared to baseline methods.

7. Acknowledgments

This work was supported by Eli Lilly and Company.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. *Proceedings of the European Conference on Computer Vision*, 2014.

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

[3] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. *Proceedings of the Advances in Neural Information Processing Systems*, pages 409–415, 2001.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[5] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*46(4):44:1–44:37, Mar. 2014.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Proceedings of the Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[7] Samantha Guerriero, Barbara Caputo, and Thomas Mensink. Deep nearest class mean classifiers. *Proceedings of the International Conference on Learning Representations, Worskhop Track*, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *Proceedings of the NIPS Deep Learning and Representation Learning Workshop*, 2015.

[10] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[11] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *The National Academy of Sciences*, 114(13):3521–3526, 2017.

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

[13] Ilja Kuzborskij, Francesco Orabona, and Barbara Caputo. From n to n+ 1: Multiclass transfer incremental learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3358–3365, 2013.

[14] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[15] Viktor Losing, Barbara Hammer, and Heiko Wersing. Incremental on-line learning: A review and comparison of state-of-the-art algorithms. *Neurocomputing*, 275:1261–1274, 2018.

[16] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pages 109–165. Elsevier, 1989.

[17] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2624–2637, 2013.

[18] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. *Proceedings of the Advances Neural Information Processing Systems Workshop*, 2017.

[19] Robi Polikar, Lalita Upda, Satish S Upda, and Vasant Honavar. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 31(4):497–508, 2001.

[20] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. *Proceedings of the IEEE International Conference on Computer Vision*, pages 1320–1328, 2017.

[21] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[22] Marko Ristin, Matthieu Guillaumin, Juergen Gall, and Luc Van Gool. Incremental learning of ncm forests for large-scale image classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3654–3661, 2014.

[23] Amelie Royer and Christoph H Lampert. Classifier adaptation at prediction time. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1401–1409, 2015.

[24] Stefan Ruping. Incremental learning with support vector machines. *Proceedings of the IEEE International Conference on Data Mining*, pages 641–642, 2001.

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[26] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Proceedings of the Advances in Neural Information Processing Systems*, pages 2990–2999, 2017.

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[28] Ragav Venkatesan, Hemanth Venkateswara, Sethuraman Panchanathan, and Baoxin Li. A strategy for an uncompromising incremental learner. *arXiv preprint arXiv:1705.00744*, 2017.

[29] Geoffrey I Webb, Roy Hyde, Hong Cao, Hai Long Nguyen, and Francois Petitjean. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994, 2016.

[30] Max Welling. Herding dynamical weights to learn. *Proceedings of the International Conference on Machine Learning*, pages 1121–1128, 2009.

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