Incremental Learning Using Conditional Adversarial Networks

Ye Xiang[1] Ying Fu[1] Pan Ji[2] Hua Huang[1]
[1]Beijing Institute of Technology [2]NEC Labs America
{xiangye,fuying,huahuang}@bit.edu.cn, peterji1990@gmail.com

Abstract

Incremental learning using Deep Neural Networks (DNNs) suffers from catastrophic forgetting. Existing methods mitigate it by either storing old image examples or only updating a few fully connected layers of DNNs, which, however, requires large memory footprints or hurts the plasticity of models. In this paper, we propose a new incremental learning strategy based on conditional adversarial networks. Our new strategy allows us to use memory-efficient statistical information to store old knowledge, and fine-tune both convolutional layers and fully connected layers to consolidate new knowledge. Specifically, we propose a model consisting of three parts, i.e., a base sub-net, a generator, and a discriminator. The base sub-net works as a feature extractor which can be pre-trained on large scale datasets and shared across multiple image recognition tasks. The generator conditioned on labeled embeddings aims to construct pseudo-examples with the same distribution as the old data. The discriminator combines real-examples from new data and pseudo-examples generated from the old data distribution to learn representation for both old and new classes. Through adversarial training of the discriminator and generator, we accomplish the multiple continuous incremental learning. Comparison with the state-of-the-arts on public CIFAR-100 and CUB-200 datasets shows that our method achieves the best accuracies on both old and new classes while requiring relatively less memory storage.

1. Introduction

In many classification tasks, we often encounter the cases where novel categories emerge after model training is done. In these cases, it's highly desirable to have a learning method at hand that can incrementally train the model on new classes while still maintaining the performance on old classes. One straightforward method is to fine-tune the model on the new classes after training on old classes. This method, however, suffers from a problem called catastrophic forgetting [24, 9], *i.e.*, little knowledge of old classes is retained after fine-tuning the original model on new classes. Another naive method is to retrain the model on both old and new data. This again becomes infeasible when quick training is demanded or when the old data is simply not available any more.

To mitigate the catastrophic forgetting problem of DNNs, two strategies are often employed: (*i*) select and store a subset of old image data to mix with new image data; (*ii*) only update the fully connected layers during incremental learning. For (*i*), the distance to the mean sample of each class is usually used as the metric for sorting samples, such as in recent methods iCaRL [28] and the End-to-End incremental model [3]. This kind of operation, unfortunately, seriously weakens the performance of trained model on old classes, since variations within each class are lost. For (*ii*), only updating fully connected layers, as did in a recent work FearNet [17], can indeed prevent drastic change in DNN models during incremental learning and thus reduce the chance of catastrophic forgetting. However, it lacks the representation learning of DNN's convolutional layers and limits the plasticity of the model, which results in inferior performance on the new classes.

In view of these deficiencies, we aim at designing a system that stores old data in a more efficient way and allows more parameters to be fine-tuned in incremental class learning. Specifically, instead of storing a subset of original images, we store the statistical information (*i.e.* mean and covariance) of feature embeddings for old classes, and fine-tune partial convolution layers together with fully connected layers during incremental class learning. To this end, we propose a conditional adversarial network which consists of three parts, *i.e.*, a base sub-net, a generator, and a discriminator. The base sub-net takes images as input and outputs convolutional feature maps, which we call *real-examples*. The generator takes as input normalized embeddings sampled from random Gaussian distributions with saved data statistics and also outputs convolutional feature maps, which we call *pseudo-examples*. The real-examples and pseudo-examples are mixed together to train the discriminator which performs both multi-category classification and true/false discrimination. The discriminator can in turn participate in adversarial learning for the genera-

tor that is conditioned on discriminative embeddings. During incremental learning, old class information is retained from pseudo-examples generated by the generator conditioned on old class statistics. We alternately train generator and discriminator to achieve multiple continuous incremental learning.

In summary, our main contributions are three-fold:

(i) we propose a new incremental learning strategy using conditional generative adversarial networks, which allows efficient storage of old information and meanwhile appropriately keeps the plasticity of model;

(ii) we build a generator, which is conditioned on embeddings of old classes and produces perceptual convolutional features as pseudo-examples to replay old class information;

(iii) we construct a discriminator, which owns two heads and can produce normalized embeddings that are discriminative for multi-category classification and indiscriminative for true/false example identification.

2. Related Work

2.1. Incremental Learning

Over the years, there have been many different incremental learning models proposed in the literature, among which traditional methods gradually evolve into recent deep learning methods.

In traditional methods, incremental learning strategy is determined by the specific classifier type. Among various classifiers, SVM [2] is the most popular and has been widely applied to large scale image classification [1]. Different incremental learning strategies based on SVM are then explored. Ruping [30] utilized the mixture of old support vectors and new examples to reduce the consumption of time and memory for incremental learning. Cauwenberghs and Poggio [4] derived an immediate and effective solution for updating support vectors by retaining the Karush-Kuhn-Tucker (KKT) conditions on all previously seen data. To circumvent the tedious training procedure of SVM classifiers, the simple nearest neighbor method is also studied for incremental learning. Ristin *et al.* [29] proposed to combine the nearest neighbor classifier and random forest to form the Nearest Class Mean Forest (NCMF), and used it for incremental learning. These traditional methods have high stability and do not sharply weaken after incorporating new classes, since the feature representation is fixed. Nevertheless, the fixed and not data-specific representation can result in low plasticity and limit the performance on new classes.

In recent years, deep learning models have drawn much attention due to their state-of-the-art performance. Under this tendency, deep-learning-based incremental learning has become dominating. However, relevant studies all suffer from catastrophic forgetting, which means typical deep neural networks tend to catastrophically forget previous knowledge when new classes are added. Many of the methods on this problem have been comprehensively reviewed in [18]. Among them, there is one kind that focuses on regularization of model parameters, such as [7], [19], and [5]. [7] creatively used the sparse representation to mitigate catastrophic forgetting of old information. [19] added additional constraint to the loss function to decrease the plasticity of parameters which contribute the most to previous tasks. [5] generalized [19] with a KL-divergence-based regularization over the conditional likelihood. Contrary to the parameter regularization, Mallya *et al.* [23] studied learning the binary masks that "piggyback" on network with fixed weights. Besides, there are more methods focusing on data-specific losses. In [22], Li and Hoiem attempted to retain old knowledge by constraining the original and new networks to have similar responses on old tasks for new data, which fully eliminated the need of requiring and storing old data. This method saves computational cost a lot, but only works in restricted scenarios where new tasks share similar discriminative features with old tasks. To better reduce the catastrophic forgetting, [28] and [3] added additional distillation loss for subset of old training images, which were selected according to the class mean examples. [37] and [13] further developed [28] by tackling the problem of data imbalance between the old and new classes. [17] proposed a brain-inspired dual-memory system, in which the fast learning memory for new classes could be consolidated to long-term storage for old classes, hence realizing incremental learning. This method adopts extra unsupervised reconstruction loss for feature vectors, and stores memory-efficient statistics of vectors. However, it only fine-tunes fully connected layers on new classes and the model architecture is not end-to-end. All in all, comparing with parameter regularization, adding data-specific loss has become more popular and is shown to be more effective for mitigating catastrophic forgetting. Hence in our method, different kinds of data-specific losses are considered and used for incremental learning.

2.2. Generative Models

We resort to generative models to implement our efficient storage strategy. One of closely related works is the Generative Adversarial Networks (GANs) [10], where an adversarial training of the generator and discriminator was introduced. Radford *et al*. [27] further developed GANs to Deep Convolutional GANs (DCGANs), and added a set of constraints on the architecture of convolutional GANs to make the training stable. These strategies have been widely utilized in other adversarial networks [31, 6, 15], and we also follow similar strategies in training our adversarial networks. On the other hand, Mirza and Osindero [25] ex-

tended GANs to a conditional version, in which generator was conditioned on extra information. The extra information can be any kind, such as class labels or data from other modalities.

Some of these generative models combined with various strategies have been explored for the incremental learning [21]. One common approach was applying unconditional GANs or GANs only conditioned on class labels to generate the image examples and performing pseudo-rehearsal to mitigate the catastrophic forgetting [32, 36]. Differing from them, our method takes the GANs conditioned on labeled embedding vectors as the generator and replays the convolutional feature maps, which is more reasonable and efficient.

3. The Proposed Method

3.1. Overview

We propose a deep-learning-based incremental learning strategy and integrate representation learning and incremental classifier learning in one framework. Specifically, we follow two guiding principles in designing our new strategy: *i*) using statistics of feature embeddings to store old knowledge; *ii*) fine-tuning both convolution layers and fully connected layers to accommodate new knowledge. In contrast to existing methods [3, 17, 22, 28, 11] which only fine-tune the fully-connected layers and/or store the original images from old classes, our method has several benefits. Firstly, compared to images, feature embeddings are more discriminative and memory efficient. Secondly, the statistics of embeddings further keeps storage consumption for each class fixed as the dataset scale expands. Thirdly, the old class information can be largely preserved when combining embedding statistics with a generator to generate faithful pseudo-examples. Finally, fine-tuning both convolutional layers and fully connected layers increases the flexibility of our model in learning new classes.

Based on above consideration, we design a system that combines information extracted from new image data and old embeddings to train the discriminator consisting of both convolutional and fully connected layers. Concretely, we propose a conditional adversarial network $A = \{B, G, D\}$ for incremental learning. As shown in Figure 1, there are three parts in it, which are a base sub-net B, a generator G, and a discriminator D. The base sub-net B serves as a feature extractor which can be pre-trained on large-scale datasets. Its parameters can be shared by most classification tasks, and are thus fixed during our incremental class learning. The generator G is conditioned on embeddings sampled from statistics of old classes and is used for generating information at the same feature-level as the output of base network B. Our network design is inspired by recent success of GANs. To realize the adversarial learning as in GANs, we call the output of B and G as real-examples and pseudo-examples, respectively. The discriminator D possesses two heads: one for multi-category classification and one for true/false example recognition. Below we will introduce the generator G, discriminator D, and incremental learning strategy in detail.

3.2. Generator

In this section, we aim to design a sub-network, which takes embeddings for old classes as input and generates features that share the same distribution and size as the output of base network B.

An optional scheme could reverse the representation extraction part of D and make a symmetric decoder to reconstruct the intermediate CNN features (*i.e*. the output of B). To train this sub-network, the mean squared error (MSE) reconstruction loss can be employed, similar to auto-encoder used in [17]. However, the intermediate features we intend to obtain are spatial features, and can not be easily reconstructed by simple MSE loss. Furthermore, totally reversing the architecture of discriminator for representation extraction also comes with higher computational complexity.

We instead formulate the problem of generating features of same distribution as the maximum likelihood estimation. From [10, 27], it has been widely accepted that GANs provide an attractive alternative to maximum likelihood estimation. Hence we design a sub-network which resembles generator of GANs, and train it in an adversarial way. Within the architecture of generator, there are several fractionally-strided convolution layers, batch normalization layers and non-linear activation layers. The arrangement of batch normalization layers and selection of activation layers are according to empirical guidelines used in [27]. The rather few number of fractionally-strided convolution layers in GANs can help reduce computational complexity and improve training speed for incremental class learning.

Although our generator originates from GANs, it becomes different when applied to incremental learning. Firstly, we use labeled embeddings of old classes instead of random noises as input, which means that our generator is conditioned on discriminative feature embeddings. Secondly, we distinguish feature-level examples, instead of image-level examples. This shares similar spirits with the perceptual loss as commonly used in image transformation tasks [15, 20]. In our problem, the feature-level examples contain more perceptual information about the images and are thus more informative for the image classification tasks.

3.3. Discriminator

In this section, we describe our discriminator for both multi-category classification and adversarial learning. Specifically, we aim to design a sub-network, which possesses two functions, *i.e*., distinguishing examples of mul-

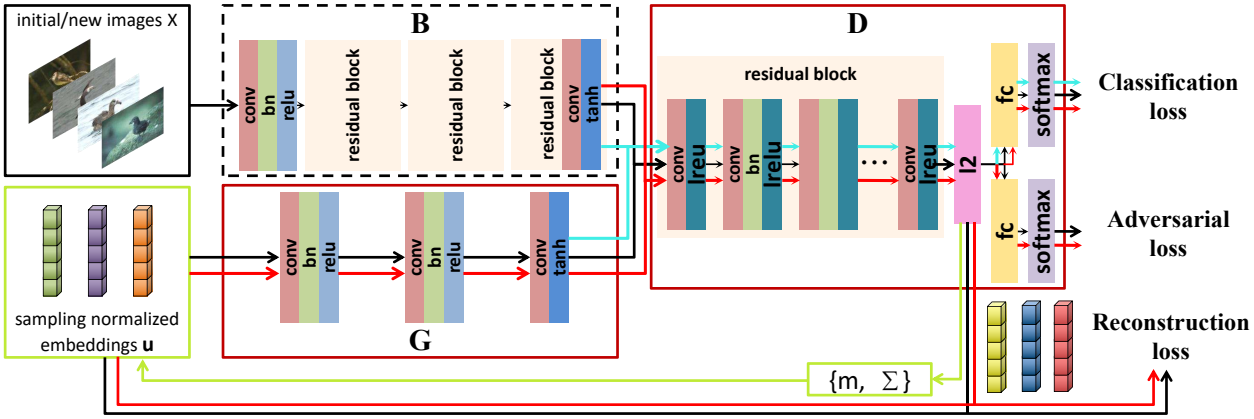

Figure 1. The architecture of our proposed model for incremental learning. It consists of three parts: a base sub-net B, a generator G, and a discriminator D with two heads. The B takes images as input and the G is fed with normalized feature embeddings sampled from statistics of old data. After adversarial training, G outputs convolutional feature representations with similar distribution to old data. During incremental learning, images from new classes are fed to B, and G is conditioned on statistics of old classes. These convolutional feature representations from both B and G are mixed together into a new mini-batch, which is then fed into D to train the classifier for both new and old classes. We take the adversarial learning strategy for our network, and alternate the training of generator and discriminator for continuous incremental learning. The cyan and red arrowed lines respectively indicate the training process for discriminator and generator.

tiple categories and assisting in adversarial learning for the generator.

To obtain the discriminative representation for multi-category classification, we inherit the feature extraction part of D from existing common CNNs [12]. That is, at certain intermediate convolution layer, we divide one typical CNN for classification into two parts, which are utilized for B and D respectively. To fulfil both two classification functions, we further add two heads on top of the feature extraction part. Note that in recent methods about semi-supervised learning with GANs [26, 34], only one extra class for recognizing pseudo-examples is added. It is because the pseudo-examples they used are unlabeled, whereas our generator is conditioned on labeled feature embeddings.

Besides, when the discriminator is participated in adversarial learning for the generator, we additionally constrain the feature embeddings output by D to be similar to input vectors of G. To reduce the matching error, we propose to add an ℓ_2 normalization layer for embeddings. Note that the output feature embedding $\mathbf{v}$ of D and the input vector $\mathbf{u}$ of G are then both the normalized unit vectors.

3.4. Loss

Our overall loss for training is

$$L = L_{adversary} + L_{classification} + L_{reconstruction}, \quad (1)$$

which consists of three different data-specific losses, *i.e.*, adversarial loss $L_{adversary}$, multi-category classification loss $L_{classification}$, and reconstruction loss $L_{reconstruction}$.

For $L_{adversary}$, we solve an adversarial min-max problem:

$$\min_{\theta_G} \max_{\theta_{D\_l}} \mathbb{E}[\log D_{\theta_{D\_l}}(B_{\theta_B}(I))] + \mathbb{E}[\log(1 - D_{\theta_{D\_l}}(G_{\theta_G}(\mathbf{u}|m_i, \Sigma_i)))], \quad (2)$$

in which I and $\mathbf{u}$ are respectively the input image for B and input embedding for G, θ_B and θ_G are parameters of B and G, θ_{D_l} means the parameters of D with the head for binary classification (*i.e.*, recognizing true/false examples), m_i and Σ_i are the mean and covariance of embeddings for class i. This is a typical optimization target for GANs, except that we take the convolution feature maps output by B to replace real image for input of D, and condition the generator G on the statistics of embedding vectors of each class.

For $L_{classification}$, we minimize the cross-entropy loss:

$$\min_{\theta_{D\_c}} -(\mathbb{E}[\log D_{\theta_{D\_c}}(B_{\theta_B}(I))] + \mathbb{E}[\log D_{\theta_{D\_c}}(G_{\theta_G}(\mathbf{u}|m_i, \Sigma_i))]), \quad (3)$$

in which θ_{D_c} represents the parameters of D with the head for multi-category classification. $L_{classification}$ constrains the representation indiscriminative for recognizing true/false examples to be discriminative for multi-category classification.

For $L_{reconstruction}$, instead of directly measuring the reconstruction of convolution feature maps by pixel-wise MSE loss, we propose to match the embeddings using cosine distance:

$$L_{reconstruction} = \mathbb{E}[1 - \cos(\mathbf{u}, D_{\theta_D}(G_{\theta_G}(\mathbf{u}|m_i, \Sigma_i)))], \quad (4)$$

in which $\mathbf{u}$ is the input embedding of G sampled from a Gaussian distribution with mean m_i and covariance Σ_i,

$D_{\theta_D}(G_{\theta_G}(\mathbf{u}))$ is the new output embedding for $\mathbf{u}$, and $\cos(\mathbf{x}, \mathbf{y})$ computes the cosine of the angle between two vectors. Furthermore, since the embedding vectors are normalized unit vectors, the cosine distance in (4) can now be equivalently re-written in the form of inner product:

$$L'_{reconstruction} = -\mathbb{E}[\mathbf{u}^T D_{\theta_D}(G_{\theta_G}(\mathbf{u}|m_i, \Sigma_i))], \quad (5)$$

3.5. Incremental Learning

Algorithm 1 Incremental Learning via Conditional Adversarial Networks

Input: Sequence of image set $\{X_0, X_1, X_2, \ldots, X_t\}$
Output: Incrementally learned model $A = \{B, G, D\}$
// Initialization
Train D using X_0;
Calculate statistics $\{m, \Sigma\}_0$ of normalized embeddings for initial classes;
Sample from statistics and get $\{\mathbf{u}\}_0$;
Alternately train G and D in an adversarial way using $\{X_0, \{\mathbf{u}\}_0\}$ and the loss in Eq. (1);
Update $\{m, \Sigma\}_0$, $\{\mathbf{u}\}_0$;
// Incremental learning
for $s = 1, \ldots, t$ **do**
    Train D using $\{X_s, \{\mathbf{u}\}_{0,1,\ldots,s-1}\}$ and the loss in Eq. (3);
    Calculate $\{m, \Sigma\}_{0,1,\ldots,s}$ and get $\{\mathbf{u}\}_{0,1,\ldots,s}$;
    Alternately train G and D in an adversarial way using $\{X_s, \{\mathbf{u}\}_{0,1,\ldots,s}\}$ and the loss in Eq. (1);
    Update $\{m, \Sigma\}_{0,1,\ldots,s}$, $\{\mathbf{u}\}_{0,1,\ldots,s}$;
end for

We study the scenario of incremental learning where samples of new classes continuously appear. This is more aligned with practical situation, unlike [22] and [16] where only transfer learning between two tasks is considered.

Given an initial image set X_0 from old classes, we first need to train an original model. In the training process, the base sub-net B which is pre-trained on the large-scale ImageNet dataset [8] is fixed, and only the discriminator D is fine-tuned. After training D, we collect the normalized embedding set from D and calculate its statistics $\{m, \Sigma\}_0$, which contains the mean vector and covariance matrix for each class. Then with the combination of X_0 and normalized embeddings $\{\mathbf{u}\}_0$ sampled from statistics, we train G and D in an adversarial way using the loss function as in Eq. (1). With the new G and D, we update statistics $\{m, \Sigma\}_0$ and corresponding sampled embeddings $\{\mathbf{u}\}_0$.

When image data of new classes X_1 arrives, we mix with the old embeddings $\{\mathbf{u}\}_0$ to retrain D. The loss function in Eq. (3) is used. Taking new images and old embeddings, we can obtain the new statistics $\{m, \Sigma\}_{0,1}$ and new sampled embeddings $\{\mathbf{u}\}_{0,1}$. New sampled embeddings together with X_1 are then mixed to alternately train G and D again using the loss in Eq. (1), and the statistics and sampled embeddings are further updated.

We repeat the above steps by recurrently training generator and discriminator every time new image data from new classes (*e.g.*, $\{X_2, \cdots, X_t\}$) appears. Then the multiple continuous incremental learning is achieved. We summarize the detailed procedure of continuous incremental learning in Algorithm 1.

4. Experimental Setup

4.1. Evaluation Metrics

To evaluate the performance of retaining old knowledge and accommodating new knowledge for different incremental learning strategies, we adopt three metrics: α_{orig}, α_{new}, and α_{all}. They respectively denote the mean test accuracies of original data, new data and all accessible data for multi-time incremental learning. For the first-time learning, *i.e.*, $t = 0$, only original data is provided and incremental learning has not truly begun, thus $\alpha_{orig,0}$, $\alpha_{new,0}$ and $\alpha_{all,0}$ are not considered. Our metrics are similar to those in [18], but we do not normalize α_{orig} and α_{all} by dividing offline model accuracy on original data. The reason is that lower accuracy of offline model can easily result in higher values of normalized metrics, which may not reflect the true accuracies.

4.2. Datasets

We employ two commonly-used public datasets for incremental learning: CIFAR-100 and CUB-200. For both two datasets, we randomly arrange the class order. The first half classes are taken to train original model, and the remaining classes are added incrementally with uniform number. We set the incremental step values utilized in out experiment to be 2, 5, 10, and 25, sequentially.

CIFAR-100 dataset consists of 100 mutually-exclusive classes, e.g., apple, baby, and house. Within each class, there are 500 images for training and 100 images for testing. All of the 60,000 images are small and have 32×32 pixels. On this dataset, images of 50 classes are initially used for training original model.

CUB-200 dataset includes 200 fine-grained bird species. For each category, the training image number is about 29-30, and the testing image number is about 11-30. There are 5,994 images for training and 5,794 images for testing in total.

4.3. Implementation Details

Our incremental learning strategy is applicable to any existing deep networks, including VGG [33], GoogLeNet [35], ResNet [12], and DenseNet [14]. In our experiments, we particularly employ ResNet-50 as the architecture of original model. This deep network has five

blocks of convolution layers. We take the first four blocks of convolution layers as the base network, and modify the last block of convolution layers for feature extraction of discriminator.

To keep the training of generator and discriminator stable, we find an important detail empirically: resetting classifier each time before fine-tuning discriminator. It is particularly effective for discriminator learning. If we continue to use previous parameters for old classes in classifier and only randomly initialize parameters for new classes, the learning for new classes just does not converge, even after adapting learning rate.

For the training of generator, we set the initial learning rate to 0.0002 and momentum to 0.5. For the training of discriminator, we set the initial learning rate to 0.05 and momentum to 0.9. For the training of both generator and discriminator, we set the weight decay to 0.0001, and epoch number to 90. After each 40 epochs, the initial learning rate is divided by 10. During all training, we use the method of stochastic gradient descent (SGD) with mini-batches to minimize predefined losses. The samples within a mini-batch are randomly and uniformly picked from the set of all image data of new classes and/or feature embeddings sampled from old class statistics.

4.4. Baseline Models

We first set a benchmark for comparison, which is an offline model trained with all required image data. We use ResNet-50 as the architecture of offline model. We also design a reference model Ours-R which is similar to our proposed model but only updates fully connected layers during incremental learning. In the Ours-R model, three fully connected layers are added after the last convolution layer of ResNet-50, and represent the discriminator. The generator and discriminator are alternately trained during continuous incremental learning.

Among existing works on incremental learning, we use several recent state-of-the-art methods based on deep CNNs as our competing baselines. They are respectively the Fixed Expansion Layer (FEL) network [7], Elastic Weight Consolidation (EWC) [19], Learning without Forgetting (LwF) [22], iCaRL [28], End-to-End incremental model [3], and FearNet [17].

5. Experimental Results

5.1. Direct Way v.s. Statistic Way of Storing Feature Embeddings

During our incremental learning, the input vectors of G are crucial for effective reconstruction of high-level perceptual information. Hence apart from the statistic way (*s-way*) we adopt, the other different approach is also explored for comparison. A natural choice is directly extracting and storing the output of ℓ_2 normalization layer introduced in Section 3.3. We call this approach *d-way*.

Table 1. Comparison of different approaches to get input vectors for G on CIFAR-100 dataset with the incremental step value as 10 classes.

		t=1	t=2	t=3	t=4	t=5
d-way	$\alpha_{orig,t}$	0.747	0.689	0.640	0.586	0.556
	$\alpha_{new,t}$	0.853	0.842	0.815	0.822	0.815
	$\alpha_{all,t}$	0.765	0.711	0.673	0.619	0.590
s-way	$\alpha_{orig,t}$	0.744	0.689	0.638	0.590	0.557
	$\alpha_{new,t}$	0.853	0.847	0.812	0.820	0.816
	$\alpha_{all,t}$	0.762	0.713	0.670	0.618	0.591

By using CIFAR-100 dataset and incremental step value of 10 classes, we compare the test accuracies for 5 times continuous incremental learning, *i.e.*, $\alpha_{orig,t}$, $\alpha_{new,t}$, and $\alpha_{all,t}$, where t ranges from 1 to 5. The results are shown in Table 1. We can see that the *d-way* and *s-way* have similar performance. However, the *s-way* can keep storage consumption fixed for each class, even when the dataset scale increases. Hence we will use the *s-way* for the following experiments.

5.2. Comparison of Accuracies

We compare the three kinds of mean test accuracies introduced in Section 4.1 between our method and other baselines on CIFAR-100 and CUB-200 datasets. The comparison results with different incremental step values are shown in Tables 2 and 3.

Across different incremental step values, we can see that the average accuracy for original data α_{orig} gradually increases. It is because the total times of incremental learning decrease, and the chance of forgetting old knowledge becomes smaller. Meanwhile, the average accuracy for new data α_{new} continuously decreases, since the classes to be incrementally learned in one time are becoming more and tasks get more difficult. For all classes seen so far, the average accuracy α_{all} is influenced more by α_{orig}, and keeps ascending slowly, or sometimes declining slightly.

In both tables, since the benchmark is trained at one time on all required images, its test accuracy is the highest and serves as an upper bound for all incremental learning methods. To properly compare the performance of different methods for incremental learning, we mainly consider α_{all} within the three different metrics. The reason is that α_{all} reflects the overall performance of the classifiers in recognizing all classes seen so far. By comparing α_{all} among various methods for incremental learning, we can clearly see that our method achieves the best performance on both datasets.

Below we analyze the performance of different methods in terms of each metric α_{orig}, α_{new}, and α_{all}. For α_{orig}, the last five methods including iCaRL, End-to-End, FearNet, Ours-R, and Ours obviously perform better than the

first three methods including FEL, EWC, and LwF. The reason is that the last five methods all store and take advantage of old data or corresponding specific perceptual knowledge during incremental learning, while the first three methods only use information extracted from previous models to retain old knowledge. Hence the data-specific information seems more effective for retaining knowledge than trained model. For α_{new}, the first three methods turn to have comparable results as other ones. It further reflects how severe the catastrophic forgetting problem is. Besides, the FearNet which only fine-tunes fully connected layers during incremental learning is not far behind for α_{new} as expected. It may be due to the use of a specialized extra module for classification of new data. For α_{all}, it can be seen that iCaRL and End-to-End perform similarly due to their similar learning strategies. FearNet and Ours-R both contain a generative model which is comprised of fully connected layers and behave more reliably than other methods at most times, but are defeated by our method, since we add extra convolution layers to better accommodate the new knowledge. All in all, there usually exists a discrepancy between retaining old knowledge and accommodating new knowledge, which requires us to find a trade-off to achieve the best comprehensive result.

We also illustrate the change of test accuracy for all classes seen so far, *i.e.*, $\alpha_{all,t}$, along with new classes continuously appearing. The two public datasets and different methods shown in Tables 2 and 3 are used for comparison. We can see from the Figure 2 that $\alpha_{all,t}$ gradually descends as new classes are continuously added for all incremental learning methods (excluding Benchmark). Among different methods, ours falls at the slowest pace, which implies the best capacity for mitigating catastrophic forgetting. Across different datasets, we can observe that methods on CUB-200 generally decline slower than on CIFAR-100. This is because the initial class number of CUB-200 is twice over that of CIFAR-100, and the original model learned with initial data owns more robust information. This is especially important for reliable learning of our generator, and leads to the greater advantage of our method on CUB-200 dataset. In view of this, our method shows great promise in applying to large scale dataset for which the initial class number is normally bigger.

5.3. Comparison of Storage

Apart from test accuracies, we also compare the storage for both data from old classes and parameters between different methods. This is particularly important for incremental learning, since if we do not consider the storage problem, an offline model could be directly adopted and there would be no need to explore other methods. Several recent methods including iCaRL, End-to-End, FearNet, and Ours-R are compared with our method.

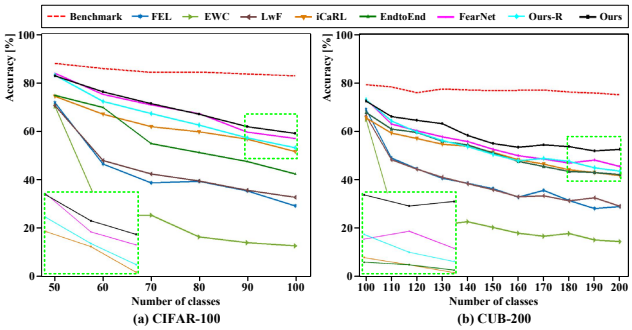

Figure 2. Test accuracies of all classes seen so far.

Before analyzing the storage of each method, let's denote the image size [1] as l, the number of images stored for each class as n, the number of classes as k, and the dimension of feature embeddings as d. Among these parameters, l, n and d are constant while k increases as incremental learning continues. For iCaRL and End-to-End which both store subsets of images from old classes, their data storage consumption is $O(3 \times l^2 \times n \times k)$. They further store feature embeddings for the additional distillation loss, which takes $O(n \times k^2)$ storage. So the total data storage for iCaRL and End-to-End is $O(3 \times l^2 \times n \times k + n \times k^2)$, and there are no auxiliary parameters besides backbone CNN. For FearNet, Ours-R and our method which store the statistics of feature embeddings and conduct pseudo-rehearsal, the data storage is $O(d^2 \times k)$, the auxiliary parameter storage is depicted by the adopted shallow network for generator which only contains 2 or 3 convolution (or fully connected) layers.

From the analysis above, we can see that the latter methods including ours have advantages over the previous ones (*i.e.*, iCaRL and End-to-End) in terms of data storage when the l and n are large, which is a typical requirement of the previous methods to achieve stable incremental learning on large datasets. Moreover, with reasonable auxiliary parameter storage, the methods that store feature embeddings can perform even better than methods that store subsets of image data. As such, the strategy of storing feature embeddings is recommended. To sum up, our method uses bounded storage for old class statistics and auxiliary generator parameters, and achieves the state-of-art performance.

5.4. Ablation Study

We conduct an ablation study for the loss function of proposed model. The public CIFAR-100 dataset with the incremental step value as 10 classes is employed.

Ours without Adversarial Loss: We remove the adversarial loss from our overall loss function. Specifically, the second head in discriminator D for recognizing true/false examples is deleted, and only classification loss and reconstruction loss are utilized. During the training of genera-

[1] Without loss of generality, we assume images are square.

Table 2. Comparison of test accuracies between our method and other baselines on CIFAR-100 dataset.

incremental step	2			5			10			25		
	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}
Benchmark	0.828											
FEL	0.316	0.759	0.330	0.351	0.732	0.373	0.362	0.707	0.378	0.364	0.684	0.374
EWC	0.102	0.750	0.154	0.144	0.731	0.179	0.163	0.715	0.187	0.226	0.676	0.240
LwF	0.306	0.817	0.328	0.362	0.735	0.379	0.381	0.714	0.396	0.395	0.675	0.423
iCaRL	0.506	0.804	0.528	0.554	0.737	0.572	0.571	0.704	0.595	0.569	0.673	0.609
End-to-End	0.486	0.882	0.503	0.459	0.832	0.497	0.485	0.791	0.532	0.507	0.696	0.567
FearNet	0.547	0.871	0.569	0.613	0.835	0.625	0.648	0.824	0.662	0.648	0.689	0.663
Ours-R	0.543	0.824	0.556	0.587	0.782	0.595	0.614	0.751	0.625	0.622	0.669	0.634
Ours	0.562	0.882	**0.580**	0.619	0.843	**0.631**	0.644	0.830	**0.671**	0.651	0.717	**0.670**

Table 3. Comparison of test accuracies between our method and other baselines on CUB-200 dataset.

incremental step	2			5			10			25		
	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}	α_{orig}	α_{new}	α_{all}
Benchmark	0.755											
FEL	0.304	0.707	0.312	0.335	0.585	0.364	0.340	0.531	0.372	0.361	0.532	0.405
EWC	0.106	0.696	0.159	0.127	0.578	0.182	0.153	0.526	0.205	0.201	0.528	0.224
LwF	0.282	0.708	0.298	0.334	0.575	0.361	0.342	0.533	0.373	0.365	0.530	0.409
iCaRL	0.400	0.715	0.443	0.473	0.594	0.496	0.495	0.529	0.505	0.503	0.532	0.514
End-to-End	0.389	0.760	0.445	0.466	0.652	0.499	0.491	0.560	0.509	0.502	0.546	0.517
FearNet	0.440	0.751	0.478	0.504	0.662	0.527	0.553	0.598	0.533	0.556	0.571	0.560
Ours-R	0.416	0.723	0.442	0.482	0.628	0.501	0.518	0.594	0.520	0.528	0.550	0.534
Ours	0.468	0.769	**0.497**	0.527	0.683	**0.549**	0.562	0.624	**0.576**	0.565	0.590	**0.576**

tor G, instead of using original adversarial manner, we just keep D fixed.

Ours without Reconstruction Loss: We remove the reconstruction loss from our overall loss function. Specifically, we omit the ℓ_2 normalization layer in D, since reducing the matching error between feature embeddings is no longer concerned. Similarly, the sampled embeddings for input of G are also unnormalized. Therefore, only the classification loss and adversarial loss are employed.

Ours with Individual Classification Loss: We remove both the adversarial loss and reconstruction loss from our overall loss function, which means only an individual classification loss is employed. Specifically, the second head in D for recognizing true/false examples and ℓ_2 normalization layer are deleted, and hence, the sampled embeddings for input of G are unnormalized. During incremental learning, we alternately fine-tune G and D by fixing one of them.

The comparison results are shown in Table 4. We can observe that both the extra adversarial loss and reconstruction loss have contributed to the improvement of accuracies. If only an individual classification loss is used, the α_{orig} and α_{all} both drop dramatically. Besides, the adversarial loss plays a more important role than reconstruction loss for α_{orig}, and is thus more capable of retaining old knowledge.

Table 4. Ablation study for loss function on CIFAR-100 dataset with the incremental step value as 10 classes. 'noAdvLoss' represents the method without adversarial loss, 'noRecLoss' means the method without reconstruction loss, and 'IndClasLoss' refers to the method with only an individual classification loss.

	Ours	noAdvLoss	noRecLoss	IndClasLoss
α_{orig}	0.644	0.623	0.635	0.597
α_{new}	0.830	0.838	0.829	0.842
α_{all}	**0.671**	0.654	0.663	0.628

6. Conclusion

In this paper, we have proposed a new strategy for incremental learning based on conditional adversarial networks. It employs statistics of normalized feature embeddings to store old knowledge, and fine-tunes both convolution layers and fully connected layers to learn new knowledge. This strategy is implemented through training a conditional adversarial network which is comprised of three parts: a base network, a generator, and a discriminator with two heads. The statistics of feature embeddings from the discriminator for the old classes are stored, and taken for obtaining input vectors of generator during incremental learning. Generator can produce pseudo-examples, which are then mixed with the real-examples output by base network and fed into the trainable discriminator. We alternately train generator and discriminator for continuous incremental learning. Comparison results on both classification accuracies and storage resources have proved the state-of-the-art performance of our method.

7. Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61425013.

References

[1] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. Good practice in large-scale learning for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(3):507–520, 2014. 2

[2] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery (DMKD)*, 2(2):121–167, 1998. 2

[3] Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 3, 6

[4] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. In *Conference on Neural Information Processing Systems (NIPS)*, 2000. 2

[5] Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *Conference on Neural Information Processing Systems (NIPS)*, 2016. 2

[7] Robert Coop, AaronMishtal, and Itamar Arel. Ensemble learning in fixed expansion layer networks for mitigating catastrophic forgetting. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 24(10):1623–1634, 2013. 2, 6

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 5

[9] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013. 1

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Conference on Neural Information Processing Systems (NIPS)*, 2014. 2, 3

[11] Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. 3

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4, 5

[13] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[14] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[15] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016. 2, 3

[16] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016. 5

[17] Ronald Kemker and Christopher Kanan. FearNet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 2, 3, 6

[18] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L. Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2, 5

[19] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*, 114(13):3521–3526, 2017. 2, 6

[20] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[21] Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. *arXiv preprint arXiv:1812.09111*, 2018. 3

[22] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(12):2935–2947, 2018. 2, 3, 5, 6

[23] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[24] 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. 1

[25] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[26] Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016. 4

[27] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2, 3

[28] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classi-

fier and representation learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 3, 6

[29] Marko Ristin, Matthieu Guillaumin, Juergen Gall, and Luc Van Gool. Incremental learning of NCM forests for large-scale image classification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[30] Stefan Rüping. Incremental learning with support vector machines. In *International Conference on Data Mining (ICDM)*, 2001. 2

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Conference on Neural Information Processing Systems (NIPS)*, 2016. 2

[32] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Conference on Neural Information Processing Systems (NIPS)*, 2017. 3

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[34] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2016. 4

[35] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5

[36] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay GANs: Learning to generate images from new categories without forgetting. In *Conference on Neural Information Processing Systems (NIPS)*, 2018. 3

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