Scalable Learning with Incremental Probabilistic PCA

Boshi Wang
Statistics Department
Florida State university
Tallahassee, FL 32306, USA
bw16d3@fsu.edu

Adrian Barbu
Statistics Department
Florida State university
Tallahassee, FL 32306, USA
abarbu@fsu.edu

***Abstract*—Incremental class learning is the classification problem of learning a model where instances from new object classes are added sequentially, and it is desired that the model be retrained only on the new classes with minimal training on the old classes. One major problem facing class incremental learning is catastrophic forgetting, where the updated model forgets the old classes and focuses only on the new classes. This paper proposes a simple and novel incremental class learning method that uses a self-supervised pretrained feature extractor to obtain meaningful features and trains Probabilistic PCA models on the extracted features for each class separately. The Mahalanobis distance is used to obtain the classification result, and an equivalent equation is derived to make the approach computationally affordable. Experiments on standard and large datasets show that the proposed approach outperforms existing state of the art incremental learning methods by a large margin. The fact that the model is trained on each class separately makes it applicable to training on very large datasets such as the whole ImageNet with more than 10,000 classes.**

***Index Terms*—large scale learning, class incremental learning, catastrophic forgetting**

I. Introduction

Incremental learning is a learning paradigm that allows a model to be updated to handle new tasks (e.g. new classes in classification) by being trained on new data without using the whole dataset. This paradigm facilitates training models on large scale datasets with less computational cost and memory requirement. A conventional approach of incremental learning is to fine-tune a pretrained model on the new data. The fine-tuning approach suffers a serious problem called catastrophic forgetting, which means that the model trained on the new classes will have a drastic performance drop on the old classes.

Various approaches have been proposed to overcome the catastrophic forgetting. Learning without forgetting (Lwf) [9] introduces the distillation loss that encourages the feature extractor to generate consistent features for the old classes. This pioneer design became an essential component in subsequent approaches. The iCarl [12] method keeps exemplars of the old classes to preserve the knowledge of these classes. Despite all the efforts to mitigate the effect of catastrophic forgetting, the overall performance remains significantly inferior to those obtained by joint training the entire dataset.

In this paper, we explore a new incremental learning strategy for multi-class image classification, using a standard pretrained feature extractor and a novel method to classify the extracted features. Assuming that the feature extractor can obtain meaningful features that don't need retraining, there is still a major problem that needs to be addressed for conventional neural networks. The problem is that the conventional classifier is based on standard projection based neurons that have high responses on half of the feature space. For this reason, it is quite likely that instances from a new class will have high responses on an old class neuron, and this issue has to be mitigated by retraining that neuron.

The RBF neuron [1] in principle alleviates this problem because an RBF neuron $f(\mathbf{x}) = g(\|bx - \boldsymbol{\mu}\|)$ will only have a high response for data near the neuron's center $\boldsymbol{\mu}$. However, experiments will show that the RBF neuron is too simple to properly handle the complexities of real data in the feature space.

For this reason, we introduce a new type of neuron that is a refinement of the RBF neuron and is based on the Probabilistic PCA model. This neuron will have high responses close to a low dimensional subspace near a neuron center $\boldsymbol{\mu}$. Experiments will reveal that this representation is capable to handle the complexities of real data, obtaining state of the art incremental learning results on standard datasets. It even allows us to obtain a reasonable classification result on ImageNet-10k, the subset of ImageNet [2] containing all the classes that have at least 450 training observations, for a total of 10,450 classes and 11 million observations.

Our paper innovates with: (1) uses a pretrained model to generate consistent feature in the whole training process; (2) introduces a PPCA based classifier to model the complexity of classes instead of standard neurons; (3) obtains an efficient computation equation for the PPCA score to make it applicable to classification on large datasets.

II. Related Work

There are two main types of approaches to class incremental learning: regularization based approaches and bias correction approaches.

A. Regularization Based Approaches

To mitigate catastrophic forgetting, regularization methods apply the distillation loss as a regularization term along with

the classification loss. This technique aims to encourage the feature extractor to generate consistent features. The distillation loss, introduced by Hinton et. al. [6], was originally used to encourage the outputs of a student network to approximate the outputs of the teacher network. In class incremental learning, the distillation loss is used as a penalty for changes in the features of old classes.

One pioneering work is Learning without Forgetting [9] (LwF), which proposed to use the distillation loss to generate stable feature representations. When the model is introduced with a new task, LwF will add task-specific parameters to the model for the new task. Despite the fact that LwF was originally proposed for task incremental learning, the distillation loss and task specific parameters have become essential components in many class incremental learning approaches.

Compared to LwF, the Incremental Classifier and Representation Learning (iCaRL) [12] method goes a step further to preserve the knowledge of old classes. iCarl also takes advantage of the distillation loss and task specific parameters. Moreover, iCarl proposes to use a limited set of exemplars to store representative samples. iCaRL also builds a dynamical mechanism to update exemplars after each training stage.

Learning without memorizing (LwM) [3] utilizes an attention-based method proposed by [15]. In order to prevent catastrophic forgetting, the attention used by the network trained on the previous tasks should not change while training the new tasks. With this restriction, features of a certain class are expected to change less when the model is trained with new classes. Different from the previous approaches, LwM takes the gradient flow information into account. By combining the distillation loss (LD) and attention distillation loss (LAD), the attention map transfers the knowledge without requiring data from the base classes during training.

To avoid catastrophic forgetting, the proposed Incremental PPCA method adopts a different approach. Instead of using the distillation loss, Incremental PPCA uses a self-supervised feature extractor pretrained on a large dataset, which is frozen during the whole training process. This simple method makes sure that the feature extractor can generate consistent features across all learning tasks.

B. Bias Correction Approaches

Bias-correction methods aim to address the problem of task recency bias. The bias is caused by the fact that the model will see more examples from the new classes than the old classes.

Hou et. al. [7] reveal that the imbalance between old classes and new classes is the main challenge causing catastrophic forgetting. In their Learning a Unified Classifier Incrementally via Rebalancing (UCIR) method, they propose to replace the standard softmax layer with a cosine normalization layer.

Wu et. al. [14] discovered that the last fully connected layer of a CNN has a strong bias towards the new classes. They proposed a method called Bias Correction (BiC), which adds an additional layer to correct the task bias of the model. BiC divides the training process into two stages. The training data is split into a training set for the first stage and a validation set for the second stage. The validation set is used to help estimate the bias in the FC layer. The earlier mentioned approach iCarl [12] doesn't use a neural network based classifier. Instead iCaRL uses the Nearest Mean Exemplar for classification. For each class, there will be a mean feature that is computed by averaging the feature representation of the exemplar images. The classification result is determined by the euclidean distance with the mean features of each class. This approach is therefore similar to the RBF neurons (which have responses based on learned centers $\boldsymbol{\mu}_i$) that are evaluated in our experiments. Our experiments reveal that an Euclidean distance based approach is too simple to capture the complexities of real data, and the proposed PPCA approach based on Mahalanobis distance is better at capturing the intrinsic data variability and has good generalization.

C. Discussion

To avoid the effect of catastrophic forgetting, the regularization based methods focus on preserving the knowledge of old classes while bias correction methods focus on improving the classifier. In these approaches, the memory buffer to store the representative examples, the distillation loss based regularization and a learning system to balance old and new classes have become three essential components for class incremental learning.

Different from previous approaches, the proposed method applies a pretrained feature extractor to obtain a consistent feature representation. Probabilistic Principal Component Analysis (PPCA) [13] models are used to approximate the complexity of the extracted features for each class. The classification scores of the extracted feature vector of an image are related to the log-likelihoods of the PPCA distribution of the classes. These scores don't change when more classes are added, therefore catastrophic forgetting does not occur in this case.

III. Proposed Method

In this section, we describe the proposed Incremental PPCA method and explain how it facilitates Incremental Learning.

Section III-A gives a formal definition of Incremental Learning. Section III-B delivers a overview of the Incremental PPCA architecture. Section III-C explains how the representative features are extracted, Section III-D states the underlying assumption about the data and obtains the Probabilistic PCA model. Section III-E details the training and Section III-F explains how the model can be sped-up for computation efficiency.

A. Problem Definition

More formally, an incremental learning problem τ consists of a sequence of T tasks:

$$\tau = [(C^1, D^1), (C^2, D^2), ...(C^T, D^T)] \tag{1}$$

where each task t is represented by a set of classes $C^t = \{c_1^t, c_2^t, ...c_{K_t}^t\}$ and training data D^t.

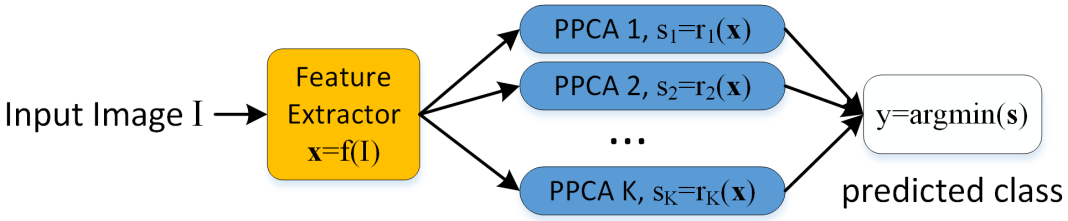

Fig. 1. Diagram of the proposed Incremental PPCA classification method.

We consider the class incremental classification problems in which the classes C^t are disjoint for each task. Let $D^t = \{(I_1, y_1), (I_2, y_2), ...(I_{n_t}, y_{n_t})\}$, where I_k are the input images and $y_k \in C^t$ are the corresponding ground truth labels.

B. Incremental PPCA Architecture

The Incremental PPCA can be written as $y = \text{argmin}_k\, r_k(f(\mathbf{I}))$, consisting of two main components: feature extractor f and classifier r. We interpret the convolutional part of a CNN model as a feature extractor $f : \Omega \to \mathbb{R}^d$, where Ω is the space of input images. Incremental PPCA takes a frozen convolutional section of a model pretrained on a large scale dataset as a feature extractor. The underlying design and choice of feature extraction are detailed in Section III-C.

Instead of using fully connected layers for classification, the classifier is designed based on the idea of Probabilistic Principal Component Analysis (PPCA). In each training stage, Incremental PPCA encodes the information about each classes in a separate model with parameters $\boldsymbol{\theta}_k$ for class k. In the inference stage, the classification results are decided by the log-likelihood scores of each image to each PPCA class model. In practice, a more computationally efficient classification equation is used to avoid large computational expenses involved in multiplying with a full covariance matrix. The underlying assumption and theoretical framework are detailed in Section III-D.

C. Feature extraction

As a major obstacle for Incremental Learning, catastrophic forgetting happens because the information about the new classes overwhelms the information about the old classes stored in the model. Hou et al [7] reveal that the imbalance between old classes and new classes is the main challenge causing catastrophic forgetting. A common underlying solution is to encourage the feature extractor to generate consistent features in the process of training with new classes. LwF [9] use the distillation loss as a regularization term. This pioneer loss design became a popular approach to avoid catastrophic forgetting. iCarl [12] goes on step further by introducing exemplars to store the information about the old classes. Yet even with these efforts, the performance of existing incremental learning methods falls behind by a large margin compared to the joint training of old and new classes.

To overcome catastrophic forgetting, the feature extractor needs to generate consistent features for the old classes when training with new classes. The choice of feature extraction follows a simple and intuitive solution: if the feature extractor is frozen in the training stage, the features of the old classes will remain consistent.

Incremental PPCA adopts as a feature extractor a CNN pretrained on a large dataset in a self-supervised manner. This feature extractor remains frozen at all times during the class incremental training process. The features are generated as $\mathbf{x} = f(I)$ where $\mathbf{x} \in \mathbb{R}^d$ is the feature extracted as a d dimensional vector from the image $I \in \Omega$ using the feature extractor $f : \Omega \to \mathbb{R}^d$.

The underlying assumption supporting this design is that with proper training, the feature extractor is able to generate features that are invariant enough to different transformations such as rotation, translation and scaling, yet contain enough information about objects without training on a specific dataset.

The experimental results indicate that an approach based on feature extraction can obtain a good test accuracy. Compared to previous approaches, training the proposed Incremental PPCA is computationally efficient because the feature extractor is frozen and the model for each class is trained separately on the observations from that class and does not need retraining when more classes are added. In experiments, we will also explore the influence of other essential factors such as image preprocessing and the model complexity.

D. Probabilistic PCA Classification

To classify instances $\mathbf{x} = f(I) \in \mathbb{R}^d$ in the feature space, we will use a Probabilistic Principal Component Analysis (PPCA) [13] for each class. PPCA has the capability to model instances for one class using a low dimensional representation that is localized near a center $\boldsymbol{\mu}$.

In the framework of PPCA, a latent variable model seeks to relate a d-dimensional observation vector $\mathbf{x}$ to a corresponding q-dimensional vector of latent (or unobserved) variables $\mathbf{t}$.

$$\mathbf{x} = \mathbf{W}\mathbf{t} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{2}$$

The $d \times q$ matrix $\mathbf{W}$ relates the two sets of variable. $\boldsymbol{\mu} \in \mathbb{R}^d$ represents the nonzero mean of observations. The latent vector $\mathbf{t} \sim \mathcal{N}(0, \mathbf{I}_q)$ contains i.i.d. Gaussians with unit variance, while $\boldsymbol{\epsilon}$ represents Gaussian noise, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$, so the probability distribution of $\mathbf{x}$ given latent variable $\mathbf{t}$ is

$$\mathbf{x}|\mathbf{t} \sim \mathcal{N}(\mathbf{W}\mathbf{t} + \boldsymbol{\mu}, \sigma^2\mathbf{I}_d) \tag{3}$$

By integration of the the latent variable $\mathbf{t}$, the marginal distribution of $\mathbf{x}$ is

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\Sigma} = \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}_d. \tag{4}$$

Using Equation (4), we can characterize each class as a Gaussian with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. Thus the classifiers of Incremental PPCA use the likelihood

$$p(\mathbf{x}|y=k) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)) \tag{5}$$

which in log terms, without the common factor $(2\pi)^{d/2}$, is simplified to the class k score:

$$s_k(\mathbf{x}) = -2\log p(\mathbf{x}|y=k) = \log|\boldsymbol{\Sigma}_k| + (\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k), \tag{6}$$

where a smaller value is better.

In practice, we will use a simpler score (the Mahalanobis distance)

$$r_k(\mathbf{x}) = (\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k) \geq 0, \tag{7}$$

which differs from $s_k(\mathbf{x})$ from Eq. (6) by the log determinant term $\log|\boldsymbol{\Sigma}_k|$. We observed that slightly better results are obtained this way.

E. Incremental PPCA Training

Training the PPCA model for class k means finding the $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k = \mathbf{W}_k\mathbf{W}_k^T + \sigma^2\mathbf{I}_d$. This is done by standard PCA using the class k observations $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$. The mean $\boldsymbol{\mu}_k$ is:

$$\boldsymbol{\mu}_k = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i,$$

and the PPCA covariance matrix is

$$\boldsymbol{\Sigma}_k = \mathbf{L}\mathbf{D}\mathbf{L}^T + \lambda\mathbf{I}_d, \tag{8}$$

where $\lambda > 0$ is a small number (e.g. $\lambda = 0.01$ in our experiments) and $\mathbf{L}$ consists of the first $q < d$ columns of $\mathbf{V}$, where

$$\mathbf{V}\mathbf{D}\mathbf{V}^T = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu}_k)(\mathbf{x}_i-\boldsymbol{\mu}_k)^T \tag{9}$$

is the SVD decomposition of the sample covariance matrix.

The parameter q represents the dimension of the linear subspace that models the variability of the class k observations. We will use the same value of q for all classes and experiments will show how the choice of q influences the model performance.

F. Efficient Computation

When d is large (e.g. $d = 1000$), computing the score for each observation involves multiplication with a large $d \times d$ matrix, which can be expensive.

Fortunately, denoting by $\mathbf{d} \in \mathbb{R}^q$ the vector containing the first q elements of the diagonal matrix $\mathbf{D}$ from Eq. (9), the score (7) can be computed faster using the following

Theorem 1. *The score* (7) *can also be written as:*

$$r(\mathbf{x}) = \|\mathbf{x}-\boldsymbol{\mu}\|^2/\lambda - \|\mathbf{u}(\mathbf{x})\|^2/\lambda, \tag{10}$$

where $\mathbf{u}(\mathbf{x}) = \operatorname{diag}(\frac{\sqrt{\mathbf{d}}}{\sqrt{\mathbf{d}+\lambda\mathbf{1}_q}})\mathbf{L}^T(\mathbf{x}-\boldsymbol{\mu})$*, and the determinant is:*

$$\log|\boldsymbol{\Sigma}| = (d-q)\log\lambda + \sum_{i=1}^{q}\log(d_i+\lambda). \tag{11}$$

Here $\operatorname{diag}(\mathbf{v})$ constructs a square matrix with diagonal elements $\mathbf{v}$, and $\sqrt{\mathbf{v}}$ for a vector $\mathbf{v}$ is computed element-wise. Observe that computing $r(\mathbf{x})$ using Eq. (10) could be 10-100 times faster than (7) since it only involves multiplication with the $q \times d$ skinny matrix $\mathbf{L}^T$ where q is usually $10-100$ times smaller than d.

Proof. We have

$$\begin{aligned}\mathbf{V}^T\boldsymbol{\Sigma}\mathbf{V} &= \mathbf{V}^T(\mathbf{L}\mathbf{D}\mathbf{L}^T + \lambda\mathbf{I}_d)\mathbf{V} \\ &= \operatorname{diag}(\mathbf{d}, 0, ..., 0) + \lambda\mathbf{I}_d \\ &= \operatorname{diag}(\mathbf{d}+\lambda, \lambda, ..., \lambda)\end{aligned}$$

because $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}_d$ and $\mathbf{L}^T\mathbf{V} = (\mathbf{I}_q, \mathbf{0})$. Thus:

$$\begin{aligned}\mathbf{V}^T\boldsymbol{\Sigma}^{-1}\mathbf{V} &= \operatorname{diag}(\frac{1}{\mathbf{d}+\lambda}, 1/\lambda, ..., 1/\lambda) \\ &= \mathbf{I}_d/\lambda + \operatorname{diag}(\frac{1}{\mathbf{d}+\lambda} - \frac{1}{\lambda}, 0, ..., 0)\end{aligned}$$

so

$$\mathbf{V}^T\boldsymbol{\Sigma}^{-1}\mathbf{V} = \mathbf{I}_d/\lambda - \operatorname{diag}(\frac{\mathbf{d}}{\lambda(\mathbf{d}+\lambda)}, 0, ..., 0)$$

and

$$\begin{aligned}\boldsymbol{\Sigma}^{-1} &= \mathbf{V}(\mathbf{I}_d/\lambda + \operatorname{diag}(\frac{\mathbf{d}}{\lambda(\mathbf{d}+\lambda)}, 0, ..., 0))\mathbf{V}^T \\ &= \mathbf{I}_d/\lambda - \mathbf{L}\operatorname{diag}(\frac{\mathbf{d}}{\lambda(\mathbf{d}+\lambda)})\mathbf{L}^T.\end{aligned}$$

The score is now:

$$\begin{aligned}r(\mathbf{x}) =& (\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) = \frac{1}{\lambda}(\mathbf{x}-\boldsymbol{\mu})^T(\mathbf{x}-\boldsymbol{\mu}) \\ &- (\mathbf{x}-\boldsymbol{\mu})^T\mathbf{L}\operatorname{diag}(\frac{\mathbf{d}}{\lambda(\mathbf{d}+\lambda)})\mathbf{L}^T(\mathbf{x}-\boldsymbol{\mu})\end{aligned}$$

from which Eq. (10) follows. □

G. Computation Complexity

In this section we are interested in the computation complexity of the Incremental PPCA method for training and testing in terms of the number N of observations, dimension d of the feature vector, number K of classes, and number q of principal vectors. Since the computation complexity of the feature extractor is $O(N)$, it will be ignored.

For training, accumulating the K covariances (e.g. using running averages) takes $O(Nd^2)$ time. Computing the PCA for the K classes takes Kd^3, thus the total training time is $O(Nd^2 + Kd^3)$, so it is linear in the number N of training observations and the number of classes K, and cubic in the dimension d of the feature representation. In practice d is on the order $d \sim 1000 - 4000$, so computing the PCAs is fast.

TABLE I
AVERAGE INCREMENTAL ACCURACY FOR INCREMENTAL PPCA VS STATE OF THE ART. THE INCREMENTAL PPCA RESULTS ARE AVERAGES OF 5 INDEPENDENT RUNS.

	CIFAR-100		ImageNet-100		Imagenet-1K		Imagenet-10k	
	5 steps	10 steps	5 steps	10 steps	5 steps	10 steps	5 steps	10 steps
iCarl [12]	57.17	52.57	65.04	59.53	51.36	46.72	-	-
BIC [14]	56.86	53.21	68.97	65.14	45.72	44.31	-	-
UCIR(NME) [7]	63.12	60.12	68.43	66.16	61.56	59.92	-	-
UCIR(CNN) [7]	63.42	60.18	70.47	68.09	64.34	61.28	-	-
PODNet [4]	64.83	63.19	75.82	73.14	66.95	64.13	-	-
PPCA-CLIP	69.71	69.71	82.02	82.02	71.25	71.25	35.42	35.42
PPCA-SWSL	77.07	77.07	86.78	86.78	76.89	76.89	34.39	34.39
RBF-CLIP	47.60	47.60	77.58	77.58	64.84	64.84	28.83	28.83
RBF-SWSL	68.64	68.64	84.14	84.14	73.414	73.414	25.81	25.81

For testing, when the models are already trained, the score computation using Eq. (10) for each observation and each class is $O(dq)$, thus the computation time is $O(NKdq)$ for all N observations and K classes. Using Eq. (7), the computation time is $O(NKd^2)$.

IV. EXPERIMENTS

In this section, we use a standard protocol for evaluating incremental methods and compare the proposed Incremental PPCA classification accuracy to the state of the art methods. We also explain how the Incremental PPCA is implemented in detail.

A. Datasets

We apply Incremental PPCA to four image datasets. Three of the datasets have been extensively used in the incremental learning literature: CIFAR-100 [8], ImageNet-100 and ImageNet-1k (ILSVRC 2016) [2]. ImageNet-100 is a subset of ImageNet-1k with only 100 classes, randomly sampled from the original 1000. The choice of these datasets facilitates the comparison with existing incremental methods from the literature. The fourth dataset is Imagenet-10k, the subset of the whole ImageNet [2] that contains all 10,450 classes that contain at least 450 training observations.

B. Evaluation Protocol

We evaluate the proposed method using the protocol introduced by Hou et. al. [7]. We start with a model trained on a random subset containing half the classes (ie., 50 for CIFAR-100 and ImageNet-100, 500 for ImageNet-1k, 5225 for ImageNet-10k). Then the remaining classes (e.g. 50 for CIFAR-100 and ImageNet-100) are incrementally added. They are equally divided among all steps, for example we could have 5 steps of adding 10 classes each time for CIFAR-100. In this case, there are 6 learning tasks in total. The trained model is evaluated after each step on the test sets of all classes that were trained so far. To facilitate the comparison, the process is repeated 5 times and the final step accuracy is averaged into a unique score called average accuracy [12].

C. Incremental PPCA Implementation

As a feature extractor, we choose a model that has been trained in a self-supervised way on a different dataset than those evaluated. CLIP (Contrastive Language-Image Pre-Training) [11] is a CNN trained on 400 million image-text pairs obtained from the web from 500,000 text queries. The image CNN part of the CLIP model adopts the popular attention mechanism as the last layer before the classification layer. We used a pretrained modified ResNet-50 classifier called `RN50x4` from the CLIP GitHub+ package [10].

SWSL is a student model whose teacher is trained on 940 million images with 1500 hashtags matching the 1000 ImageNet synsets. The student is trained on a subset of 64 million images selected by the teacher. The SWSL model we use is a pretrained standard ResNet-50, called `resnet50_swsl` from the Facebook Research model repository on GitHub [5].

The input images are resized to 288×288 except for CIFAR-100 where they were resized to 144×144. The features were extracted before the classification layer using the classifier's attention pooling layer, except for CIFAR-100 where average pooling was used instead. The dimension of the extracted features was $d = 640$ except for CIFAR-100 where it was $d = 2560$.

All experiments were implemented using the PyTorch framework. After all the features were extracted, a separate PPCA model was trained on the training data from each class, as described in Section III-E. Observe that these models do not change no matter what or how many other classes are added. Training the PPCA models takes about 20s per class on a NVIDIA GeForce RTX3080m GPU laptop, thus about 5.6h for 1000 classes.

D. Results

We evaluate the Incremental PPCA mainly by its classification performance on the validation set of the four datasets. The classification performance is compared with four state of art models. The models deliver quite remarkable performance, as shown in Table I. Some of their main components are designed with a similar underlying philosophy with Incremental PPCA. As a baseline, BIC [14] stated that traditional Neural Networks have a strong bias towards the new classes. BIC adopts a linear model as classifier. iCarl [12] and UCIR [7] both use the Nearest-Mean-Exemplar (NME) for classification. UCIR also uses a second inference method based on probabilities (UCIR-CNN).

All the comparable models discard the neuron-based structure and explore new classification structures. So the comparison with these models can be an opportunity to evaluate the power of the PPCA based classifier.

TABLE II
AVERAGE ACCURACY OF PPCA-CLIP ON CIFAR-100 FOR DIFFERENT INPUT SIZES AND NUMBERS OF PRINCIPAL COMPONENTS.

Input size	0PC	5PC	10PC	20PC	50PC	100PC	200PC
32×32(original)	7.05	21.04	21.04	25.7	27.37	28.60	28.58
36×36 (avg pooling)	7.47	22.26	24.83	26.66	28.9	29.55	29.51
72×72 (avg pooling)	40.32	53.43	55.04	55.04	58.89	59.56	60.07
144×144 (avg pooling)	47.60	61.57	64.34	66.58	68.50	69.56	69.71
288×288 (avg pooling)	30.97	51.09	56.35	60.48	63.94	65.05	65.71
288×288 (with attention)	40.88	55.29	58.54	61.44	62.78	62.14	61.26

TABLE III
AVERAGE ACCURACY OF PPCA-SWSL ON CIFAR-100 FOR DIFFERENT INPUT SIZES AND NUMBERS OF PRINCIPAL COMPONENTS.

Input size	5PC	10PC	20PC	50PC	100PC
32×32(original)	41.62	44.14	45.4	47.17	48.41
64×64	61.93	63.4	64.72	66.08	66.63
128×128	73.53	75.02	75.97	76.89	77.07
256×256	69.32	71.68	73.51	74.98	74.92
224×224 (default)	69.03	71.17	72.8	73.72	74.43

We also compare a simpler classifier based on the nearest Euclidean distance to the observation means $\boldsymbol{\mu}_k$ instead of the Mahalanobis distance (7):

$$r_k(\mathbf{x}) = \|\mathbf{x} - \boldsymbol{\mu}_k\|^2. \tag{12}$$

This classifier is shown as RBF-CLIP and RBF-SWSL in Table I.

1) CIFAR-100.: The CIFAR-100 dataset contains images of size 32×32 pixels. The CLIP feature extractor is trained with images of size 288×288, so it will not work best with such small images even when resized to 288×288. To gain the best inference performance, experiments were designed to find the best input size for CIFAR-100. The experiments, discussed in Section V-A, reveal that the best input size is 144×144 when the feature extractor is from CLIP, the best input size is 128×128 when feature extractor is SWSL. From Table I, we can see the Incremental PPCA with CLIP obtains an average accuracy of 69.71%, outperforming the state of the art by almost 5%. SWSL gain the average accuracy of 77.07% on averaged accuracy. Surpassing the previous state of art of PODNet by 12.34%.

The RBF-CLIP experiment shows that when only the class means $\boldsymbol{\mu}_k$ are used, obtaining RBF-style neurons, the average accuracy drops drastically to 47.6% with CLIP. The average accuracy drops to 68.64% with SWSL. This result indicates that the PPCA model is capable of capturing meaningful variation in the data, much better than using Euclidean distance to the class means.

2) ImageNet-100.: ImageNet-100 contains 100 randomly selected classes from the ILSVRC2016 dataset. The images have the comparable size with the size the CLIP feature extractor has been trained on. From Table I, when the feature extractor is CLIP, the average accuracy on the ImageNet-100 validation set has reached 82.02%, surpassing the next best method (PODNet) by 6.2%. When the features extractor is SWSL, the average accuracy is 86.78%, surpassing PODNet by 10.96%

Considering that the CLIP feature extractor was not trained on ImageNet-1k or ImageNet-100, the performance of Incremental PPCA indicates that using a pretrained feature extractor can obtain meaningful features for classification. Again, the RBF-CLIP falls behind PPCA-CLIP by about 5%.

3) ImageNet-1k.: ImageNet-1k is a more challenging dataset, containing 1000 classes. The images are also comparable in size with the size the feature extractor has been trained with. From Table I, Incremental PPCA with CLIP as feature extractor has reached 71.25% average accuracy. The performance has surpassed the state of art PODNet by 4.3% and RBF-CLIP by more than 6%. Incremental PPCA with SWSL as feature extractor has reached 76.89% average accuracy. The performance has surpassed the state of art PODNet by 12.76% and RBF-CLIP by more than 3.45%.

4) ImageNet-10k.: ImageNet-10k is the most challenging dataset, containing 10,450 classes and 11 million training images. None of the other methods report results on it, probably because it is very large. From Table I, Incremental PPCA with CLIP as feature extractor obtains 35.42% average accuracy with 20 principal components. This is much better than random guessing, which would be 0.01% in this case. At the same time, the RBF-CLIP that just uses the class means $\boldsymbol{\mu}_k$ and the Euclidean distance obtains a 28.83% accuracy, a 6.59% difference.

E. Discussion

The results from Table I indicate that the proposed PPCA-CLIP method can outperform the state of the art incremental class learning methods by at least 4% on all three datasets where a comparison can be obtained. Moreover, it can obtain reasonable classification results on ImageNet-10k, a dataset with 10,450 classes and more than 11 million images.

Furthermore, the RBF-CLIP classifier that just uses the class means and Euclidean distance for classification falls behind by about 5% in accuracy. This shows that the PPCA model is capable of capturing a meaningful representation of the data that goes beyond just radial basis functions and Euclidean distance.

TABLE IV
AVERAGE ACCURACY OF PPCA-CLIP ON IMAGENET-1K AND IMAGENET-100 FOR DIFFERENT NUMBERS OF PRINCIPAL COMPONENTS.

Input size	0PC	5PC	10PC	20PC	50PC	100PC	200PC
ImageNet-100 with CLIP	77.58	80.92	81.26	82.02	81.24	79.92	78.12
ImageNet 100 with SWSL	84.14	86.22	86.78	86.58	86.72	86.76	86.78
ImageNet-1k CLIP	64.84	69.16	70.73	71.25	69.93	68.11	65.79
ImageNet with SWSL	73.41	76.46	76.81	76.89	76.76	76.57	76.04
ImageNet-10k CLIP	28.83	33.49	34.85	35.42	34.80	33.28	31.01
ImageNet-10k SWSL	25.81	31.75	33.39	34.39	34.75	34.40	33.56

V. ABLATION STUDY

In this section, we evaluate the contribution of different hyper-parameters to the proposed Incremental PPCA method's accuracy. There are two essential components of the proposed method: feature extraction and the PPCA classifier. The size of the feature extractor's input influences the quality of the extracted features. In Section V-A, we will see how the input size changes the accuracy of the Incremental PPCA method for CIFAR-100. The number of principal components (PCs) decides how much complexity is encoded for each class. Its effect will be analyzed in Section V-B.

A. The Input Image Size

As show in Table II and Table III, the PPCA accuracy varies with the size of the feature extractor input. The CLIP feature extractor is trained with high resolution (288×288) images, while the images of the CIFAR-100 dataset are 32×32. If these images are resized to 288×288, they will look very blurred. The experiment aims to explore how the input size improves the performance on low resolution images. It might as well be possible that the 288×288 input might not be the best setting for low resolution images for the high resolution feature extractor.

The evaluation of input size is based on the experiment with 5 steps of 10 classes on CIFAR-100. Table II and Table III shows the contribution of the input size to the method's accuracy. If the image are input without resizing, the method's accuracy is very poor. When is feature extractor is SWSL, the original preprocessing that resizes the images to 224×224 doesn't produce the best performance. When the 32×32 images are resized to 128×128, the average accuracy reaches a maximum of 77.07%.

When is feature extractor is CLIP, the original preprocessing that resizes the images to 288×288 doesn't produce the best performance either. When the 32×32 images are resized to 144×144, the average accuracy reaches a maximum of 69.71%. The results are also shown as a graph in Figure 2.

In Table II, the performance is also compared between the application and absence of attention pooling in the feature generation process. Attention pooling can only be used for 288×288 images. The average accuracy without attention pooling reaches the 69.71% maximum, surpassing the performance with attention pooling. The comparison indicates that attention pooling may use high resolution information in the feature extraction process, which is not present in the low resolution images.

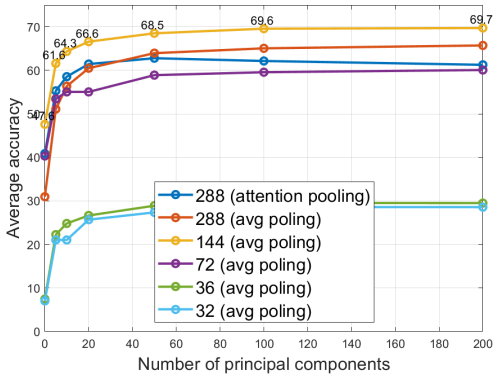

Fig. 2. Incremental PPCA with CLIP and different input sizes on CIFAR-100.

These experiments indicate that when the feature extractor trained with high resolution images is applied to low resolution images, a medium input size between high and low resolution can help classification reach optimal performance.

B. The Number of Principal Components

The number of principal components q is a essential hyper-parameter that influences the method's accuracy. Theoretically, q represents the dimension of the (linear) manifold the observations in each class belong to.

We design an experiment to explore the relation between the number of principal components and classification accuracy. Again, the evaluation is based on the experiment with 5 steps of 10 classes on CIFAR-100, ImageNet-100, ImageNet-1k and ImageNet-10k. We use 0 principal components as a baseline in which case no variation are encoded. Even though the SWSL is finetuned on ImageNet-1k, the results with SWSL can also be representative for the ResNet based model.

Figure 2 reveals that the average accuracy saturates as the number of PCs is increased for CIFAR-100, the same phenomenon for all input sizes evaluated.

Table IV and Figure 3 show that the average accuracy increases with the number of principal components until the number of PCs reaches 20, after which the average accuracy drops gradually.

These experiments reveal that an appropriate number of principal components can retain balance between overfitting and lack of variation to reach an optimal classification accuracy.

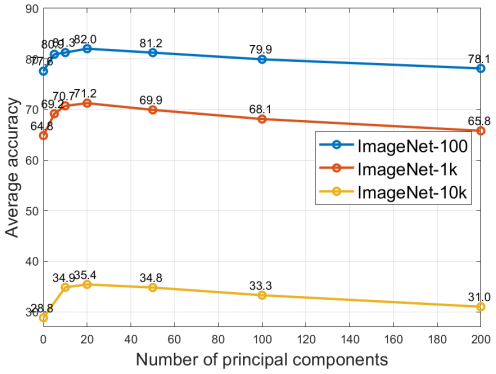

Fig. 3. Incremental PPCA with different number of principal components on ImageNet-100 and ImageNet-1k.

VI. Conclusion

This paper presented a novel and simple method for large scale incremental class learning with pretrained feature extractors and a classifier based on probabilistic PCA models. The pretrained feature extractor guarantees that the generated features are consistent during the whole training process. The PPCA based classification is able to encode the complexity of each image class using a low dimensional representation and small computational expenses. This approach can prevent catastrophic forgetting to a large extent.

The proposed method outperforms existing state of the art methods in incremental class learning on three standard datasets used in the literature. Furthermore, it obtains reasonable classification results on ImageNet-10k, the subset of ImageNet containing all classes with at least 450 training images, for a total of more than 11 million images.

The performance of Incremental PPCA reveals that generic pretrained models have the ability to extract meaningful features from images and gain promising performance without being trained on labeled data.

Humans are capable of classifying millions of classes of objects and of learning a new object class from only 1-2 examples. However, humans are the subjects of millions of years of evolution, which can be seen as a kind of supervised learning since the most fit specimens (e.g. fit for understanding images) are better at reproduction and evolution. Thus the human brain is pre-wired by evolution to be able to accomplish these large scale classification tasks. The pretrained feature extractor that we use in the Incremental PPCA can be seen as the analogue of the pre-wired brain in humans and other mammals.

In the future we plan to further extend our method for classification with hundreds of thousands of classes using hierarchical techniques, to get one step closer to the scale of human classification capabilities.

References

[1] Broomhead, D.S., Lowe, D.: Radial basis functions, multi-variable functional interpolation and adaptive networks. Tech. rep., Royal Signals and Radar Establishment Malvern (United Kingdom) (1988)

[2] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255. Ieee (2009)

[3] Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: CVPR. pp. 5138–5146 (2019)

[4] Douillard, A., Cord, M., Ollion, C., Robert, T., Valle, E.: Podnet: Pooled outputs distillation for small-tasks incremental learning. In: ECCV. pp. 86–102. Springer (2020)

[5] Facebook: SWSL. https://github.com/facebookresearch/semi-supervised-ImageNet1K-models (2022), [Online; accessed 09-02-2022]

[6] Hinton, G., Vinyals, O., Dean, J., et al.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 **2**(7) (2015)

[7] Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: CVPR. pp. 831–839 (2019)

[8] Krizhevsky, A.: Learning multiple layers of features from tiny images. Master's thesis, University of Toronto (2009)

[9] Li, Z., Hoiem, D.: Learning without forgetting. IEEE Trans. on PAMI **40**(12), 2935–2947 (2017)

[10] OpenAI: CLIP. `https://github.com/openai/CLIP` (2022), [Online; accessed 09-02-2022]

[11] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML. pp. 8748–8763 (2021)

[12] Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: CVPR. pp. 2001–2010 (2017)

[13] Tipping, M.E., Bishop, C.M.: Probabilistic principal component analysis. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **61**(3), 611–622 (1999)

[14] Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning (2019)

[15] Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. CoRR **abs/1612.03928** (2016), `http://arxiv.org/abs/1612.03928`