FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

# SEMI-SUPERVISED FEW-SHOT LEARNING

# WITH PROBABILISTIC PRINCIPAL COMPONENT ANALYZERS

By

KE HAN

A Dissertation submitted to the
Department of Statistics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

2024

Ke Han defended this dissertation on November 6, 2024.

The members of the supervisory committee were:

Adrian Barbu
Professor Directing Dissertation

Kyle Gallivan
University Representative

Wei Wu
Committee Member

Joshua Loyal
Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.

*Dedicated to Chenxi, whose unwavering support helped me through moments of struggle. To my parents, who instilled in me the values of honesty and courage, shaping the person I am today.*

# ACKNOWLEDGMENTS

I would like to begin by expressing my deepest gratitude to my advisor, Dr. Adrian Barbu, whose patient guidance and insightful advice have been instrumental throughout my research journey. This dissertation would not have been possible without his unwavering support and mentorship. I am also incredibly thankful to my committee members, Dr. Kyle Gallivan, Dr. Wei Wu, and Dr. Joshua Loyal, for their valuable feedback and thoughtful suggestions, which greatly enriched my work. I sincerely appreciate all of your help and contributions in making this dissertation a reality.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| FSL | Few-Shot Learning |
| CIL | Class Incremental Learning |
| FSCIL | Few-Shot Class Incremental Learning |
| SSFSCIL | Semi-Supervised Few-Shot Class-Incremental Learning |
| PCA | Principal Component Analysis |
| PPCA | Probabilistic Principal Component Analyzers |
| CUB200 | Caltech-UCSD Birds-200-2011 |

# ABSTRACT

Few-Shot Learning (FSL) has emerged as a pivotal area of research in machine learning, addressing the challenge of training models to recognize classes with limited labeled data. This paradigm is particularly relevant in real-world scenarios where obtaining large annotated datasets is impractical or costly. This dissertation incorporates semi-supervised learning and investigates two major advancements in the realms of Few-Shot Class Incremental Learning (FSCIL) and few-shot classification on large-scale datasets. It tackles the limitations of conventional approaches and strives to improve overall performance in these complex contexts. The second chapter introduces a novel approach to FSCIL that effectively minimizes the issue of catastrophic forgetting — a common challenge in incremental learning scenarios. This method utilizes a generic feature extractor, pretrained on a vast dataset, combined with the proposed $k$-Probabilistic Principal Component Analyzers ($k$-PPCAs) model as the classifier for a collection of classes. Unlike conventional methods that rely on fully connected layers for projection, this PPCA-based approach localizes classes around their respective means in a shared feature space, allowing for the seamless addition of new classes without the need to retrain existing models. The learning algorithm is a modified k-Means, strategically designed to freeze the models of established classes while updating only those corresponding to new classes. This dual strategy not only enhances computational efficiency but also ensures high accuracy with less catastrophic forgetting. Experimental results demonstrate that this innovative approach significantly outperforms existing FSCIL methods, particularly in tests conducted on the ImageNet-1k dataset — a large and complex dataset that has previously been underutilized in FSCIL research. Building on the insights gained from the first project, the second endeavor presents a hierarchical few-shot semi-supervised framework, Hierarchical $k$-Probabilistic Principal Component Analyzers (H$k$-PPCAs), aimed at improving few-shot classification within large-scale datasets. The H$k$-PPCAs structure the feature space in a two-level hierarchy, modeling both the first-level image classes and second-level super-classes as Gaussian distributions via PPCA. This hierarchical design not only facilitates the scalable addition of new classes without necessitating full model retraining but also significantly reduces classification time from $O(K)$ to $O(\sqrt{K})$, making the framework both efficient and practical for large-scale classification tasks. Rigorous experiments on the ImageNet-1k and ImageNet-10k datasets validate the effectiveness and robustness of this approach, showcasing its ability to maintain classification stability under few-shot conditions.

# CHAPTER 1

# INTRODUCTION

Few-Shot Learning (FSL) is a machine learning technique where a model learns to make predictions and generalize well with only a small number of training examples. Few-shot learning is an extension of unsupervised and semi-supervised learning. Unlike traditional machine learning supervised models, which rely on large amounts of labeled data for training, Few-Shot Learning focuses on learning efficiently with limited labeled examples—usually as few as five or fewer. Few-shot learning is defined as the ability of a model to learn from a few examples and generalize to new tasks. It typically relies on prior knowledge or pre-training on related tasks, enabling the model to quickly adapt to new challenges with minimal data.

Few-shot learning is targeted to addresses several challenges. Data scarcity is the situation where labeled data is scarce or expensive to obtain. Under this circumstance, Few-Shot Learning allows models to perform well even with very limited labeled examples, this advantage can significantly reduce the dependency on large labeled datasets and is crucial in fields like medicine or astronomy, where obtaining labeled data is both expensive and difficult. Traditional models usually struggle to generalize when the new tasks are quite different from the training tasks, but Few-Shot Learning can quickly adapt to different tasks by learning how to extract meaningful information from previous experience and improve the model's generalization ability when faced with new, unseen tasks. This fast adaptation to new tasks is particularly useful in dynamic environments or rapidly changing scenarios. Few-shot learning operates with a very small number of labeled samples, thus it reduces training time and computational resources, making it suitable for scenarios where hardware capacity is limited. Few-shot learning is a significant area of focus in artificial intelligence and machine learning research, particularly in solving data-scarcity problems and improving task adaptability. It holds great potential for applications in domains where acquiring large amounts of labeled data is difficult.

Few-shot learning typically involves several popular approaches. Meta-learning is often called "learning to learn", which involves training the model across many tasks so that it can quickly adapt to new tasks using only a few examples. The model extracts general patterns from diverse tasks to transfer to new ones.

Meta-learning-based methods have several popular learning strategies: metric learning, optimization-based methods, and feature transfer. Metric learning models (Vinyals et al., 2016; Snell et al., 2017; Oreshkin et al., 2018; Zhang et al., 2020) learn to measure the similarity between different data points, allowing it to make predictions on new tasks by comparing new examples to those in the few-shot set. Optimization-based methods (Ravi and Larochelle, 2016; Finn et al., 2017) focus on optimizing gradient-based model parameters efficiently with limited data. Regarding feature transfer learning methods (Santoro et al., 2016; Snell et al., 2017; Oreshkin et al., 2018; Qi et al., 2018; Chen et al., 2019), the model typically leverages and transfers the features pretrained from related tasks to improve performance on new tasks with limited data.

There has been a growing interest in leveraging unlabeled data to enhance the accuracy of Few-Shot Learning, and this interest breeds the development of Semi-Supervised Few-Shot Learning (SSFSL), which is an overlapped set of Semi-Supervised Learning (SSL) and Few-Shot Learning. Semi-Supervised learning is a type of machine learning that combines a small amount of labeled data with a large amount of unlabeled data during the training process. The goal is to leverage the structure of the unlabeled data to improve the performance of the model, making it particularly useful in scenarios of Few-Shot Learning. However, directly applying SSL methods to the few-shot setting often yields subpar results due to the extremely small number of labeled observations. A lot of endeavors (Snell et al., 2017; Ren et al., 2018; Liu et al., 2018; Wang et al., 2020; Lazarou et al., 2021; Huang et al., 2021) have been made to address the challenges of SSFSL.

In this dissertation, we utilize feature transfer and metric learning strategies and propose two novel semi-supervised few-shot learning frameworks, respectively tackled with two important challenges in Few-Shot Learning, Class Incremental Learning (CIL) and Large-scale learning.

In Chapter Two, we delve into a subbranch of Few-Shot Learning within the context of Class Incremental Learning (CIL). CIL is a rapidly evolving research area, closely tied to advancements in deep learning. The goal of CIL is to incrementally train a unified classifier capable of recognizing all the classes encountered throughout the training process. However, because labeled data is often scarce and costly to obtain, there is growing interest in maximizing the utility of the available labels. In this context, Few-Shot Class Incremental Learning (FSCIL) has emerged as an extension of CIL, where the constraint is that only a limited number of labeled samples per class are available in each incremental session.

Despite its promise, FSCIL methods face significant challenges, particularly overfitting and catastrophic forgetting, due to the small number of labeled examples. To address these issues, we propose a simple yet effective framework called $k$-Probabilistic Principal Component Analyzers ($k$-PPCAs), which is inspired by Probabilistic PCA (PPCA) and organizes the output space into a collection of PPCAs. By applying this approach to the CIL task, our method outperforms most existing state-of-the-art techniques on three widely-used FSCIL benchmarks. Additionally, we are the first to evaluate FSCIL on a large-scale benchmark, which is less commonly used due to its extensive output space. The experimental results are presented in Chapter Three.

Building on the effectiveness of the proposed method for large-scale datasets, in Chapter Four, we extend the approach to larger-scale few-shot classification by incorporating the concept of hierarchical classification. Hierarchical classification is a machine learning strategy that organizes classes or categories into a hierarchical tree structure. We propose a new framework, called Hierarchical $k$-PPCA, which leverages this hierarchy to improve classification. In this model, classes are represented using Gaussian distributions constructed through PPCAs, and a two-level taxonomy is used to reflect the semantic relationships between classes, though more levels can be added if necessary.

The method begins by initializing image classes with labeled data using PPCAs. Then, pseudo-labels are predicted for unlabeled data by identifying the nearest class using the Mahalanobis distance. The class representations are iteratively refined until convergence is achieved.

At the higher semantic level, classes that share similar characteristics are grouped into semantic clusters, referred to as super-classes, which are modeled by Gaussian distributions. A modified k-means clustering algorithm is used to uncover the underlying relationships between image classes and construct the hierarchical structure. During classification, for instance, an image of a panda could be first assigned to a super-class like 'Ursidae' and then further classified into one of the image classes within the 'Ursidae' super-class.

This hierarchical framework reduces the search space and computational requirements compared to flat classification models. For a dataset with $K$ classes, the hierarchical model can classify images in $O(\sqrt{K})$ time, a significant improvement over the $O(K)$ time required for non-hierarchical classification. Experiments detailed in Chapter Five demonstrate that the proposed method is both more accurate and efficient when applied to large-scale datasets.

In Chapter Six, we provide a summary of the conclusions drawn from this dissertation and outline several directions for future research.

# CHAPTER 2

# SEMI-SUPERVISED FEW-SHOT INCREMENTAL LEARNING WITH K-PROBABILISTIC PCAS

## 2.1 Introduction

Class Incremental Learning (CIL) is an active research topic along with the development of deep learning. It aims to incrementally learn a unified classifier to recognize all classes that have been met during training. However, due to the scarcity of costly labeled data in practice, there is an emerging interest in promoting the utility of existing labels. Against this background, the Few-Shot Class Incremental Learning (FSCIL) task (Tao et al., 2020) has recently risen from CIL. An FSCIL model needs to involve new classes with limited labeled observations sequentially without forgetting the discriminability of old classes. However, due to the limited number of labeled samples, FSCIL methods are prone to overfitting and catastrophic forgetting compared to traditional CIL.

From the FSCIL perspective, preserving representative memories (Tao et al., 2020; Zhang et al., 2021; Zhu et al., 2021; Peng et al., 2022) and knowledge distillation (Dong et al., 2021; Cui et al., 2021) are popular techniques to maintain the representation capability for old classes and address catastrophic forgetting. For mitigating the overfitting problem from the very few labeled samples, existing methods have proposed different solutions, for example by decoupling the representation and classifier (Zhang et al., 2021), and the node-adjustable network (Yang et al., 2021).

Compared to labeled data, unlabeled data is less expensive and easier to access in the real world. Hence, it is naturally used to leverage the performance in the incremental sessions. Cui et al. (2021) first proposed Semi-Supervised Few-Shot Class-Incremental Learning (SSFSCIL), a sub-task of FSCIL. They utilized exemplars and knowledge distillation to mitigate catastrophic forgetting and alleviated overfitting with supervised massive unlabeled samples. One of the other recent SSFSCIL works, FeSSSS (Ahmad et al., 2022), fuses a multi-layer perceptron with self-learning and pretrained supervised feature generators to overcome overfitting.

These neural network-based SSFSCIL methods achieved remarkable performance on different popular FSCIL benchmarks. However, the retraining of neural networks can be difficult due to several factors, e.g. continuous adapting to the data stream, data imbalance, privacy concerns, computational effort, etc. To

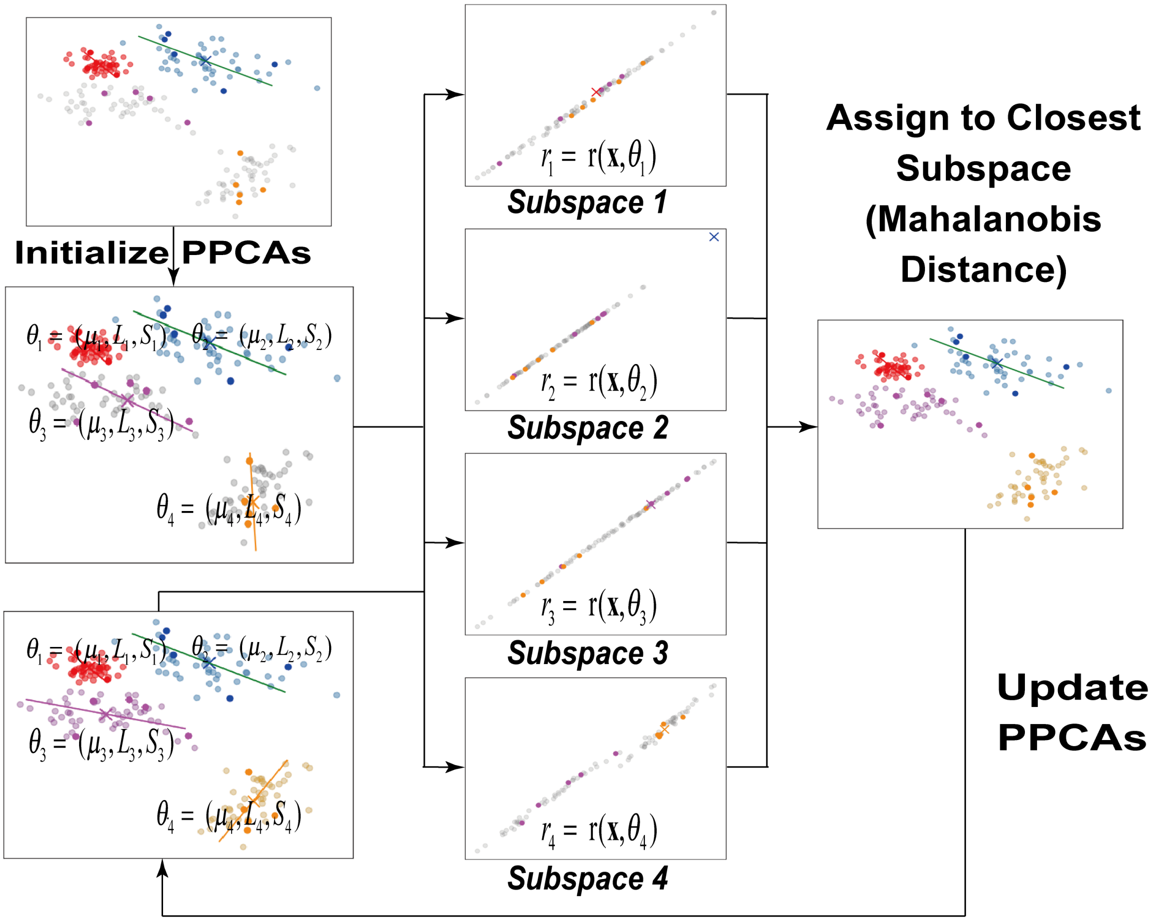
Initialize PPCAs
$\theta_1 = (\mu_1, L_1, S_1)$
$\theta_2 = (\mu_2, L_2, S_2)$
$\theta_3 = (\mu_3, L_3, S_3)$
$\theta_4 = (\mu_4, L_4, S_4)$
$\theta_1 = (\mu_1, L_1, S_1)$
$\theta_2 = (\mu_2, L_2, S_2)$
$\theta_3 = (\mu_3, L_3, S_3)$
$\theta_4 = (\mu_4, L_4, S_4)$
$r_1 = \mathrm{r}(\mathbf{x}, \theta_1)$
Subspace 1
$r_2 = \mathrm{r}(\mathbf{x}, \theta_2)$
Subspace 2
$r_3 = \mathrm{r}(\mathbf{x}, \theta_3)$
Subspace 3
$r_4 = \mathrm{r}(\mathbf{x}, \theta_4)$
Subspace 4
Assign to Closest Subspace (Mahalanobis Distance)
Update PPCAs

Figure 2.1: Flowchart of k-PPCAs (2-way 5-shot). The colored points are labeled or classified observations. The grey points are unlabeled.

reduce the inconvenience of retraining and lessen catastrophic forgetting, this Chapter proposes a novel SS-FSCIL framework, termed $k$-Probabilistic Principal Component Analyzers ($k$-PPCAs). First, the proposed method models the class embedding in the feature space with the Gaussian distribution reconstructed from a lower-dimensional subspace. Specifically, each class is modeled by an individual Probabilistic PCA (PPCA) (Tipping and Bishop, 1999b), and all classes are systematized as a collection of PPCAs, which is essentially a mixture of Gaussian models (Tipping and Bishop, 1999a). When new classes are added, the PPCA models for the old classes do not necessitate retraining, and catastrophic forgetting can be significantly alleviated. Second, a Mahalanobis distance-based $k$-Means is introduced to perform parameter estimation and classification for the FSCIL problem. This modification considers and leverages the shape information of class embeddings. Furthermore, this metric-based classifier can help identify the out-of-distribution samples. Third, the framework uses a generic pretrained feature extractor which is frozen and static during training.

The design can take advantage of the recent developments in self-learning algorithms, and the benefit of decoupling the representation learner and classifier has also been shown in mitigating catastrophic forgetting (Zhang et al., 2021; Ahmad et al., 2022; Peng et al., 2022). To demonstrate the effectiveness of the proposed method, extensive experiments are performed on four benchmark datasets: CUB200, CIFAR100, miniImageNet, and ILSVRC-2012 (ImageNet-1k). The proposed method outperforms most of the state-of-the-art SSFSCIL methods on the first three benchmarks. To the best of our knowledge, there are no SSFSCIL methods evaluated on the ImageNet-1k dataset due to its large scale, where the proposed $k$-PPCAs achieves an accuracy of 58.44%.

The main contributions of proposed method in this chapter are:

- It proposes a semi-supervised FSCIL framework named $k$-PPCAs that avoids the retraining for old classes so that catastrophic forgetting can be significantly reduced when new classes are incorporated.
- A Mahalanobis distance-based $k$-Means classifier is adapted for the FSCIL task, which can capture the shape information of class embeddings and classify the out-of-distribution samples to ”unknown” through a low-confidence score metric.
- Performs a comprehensive comparison between the proposed method and state-of-the-art semi-supervised FSCIL methods on three popular FSCIL benchmarks: CUB200, CIFAR100, and miniImageNet. The results show that the proposed method outperforms most of the other methods.
- Conducts experiments on the large-scale dataset ImageNet-1k, which is less frequently evaluated for the FSCIL task.

## 2.2 Related Work

Because FSCIL is a subdomain of class-incremental learning, this section starts with the related works in class-incremental learning.

**Class-Incremental Learning (CIL).** Class-incremental learning aims to learn a unified classifier to recognize new incremental classes without forgetting the old class representations. There are two popular ways to mitigate catastrophic forgetting, exploring better exemplars for old classes and modifying loss function. iCaRL(Rebuffi et al., 2017) dynamically updates important exemplars and adopts nearest-class-mean(Mensink et al., 2013) for classification. Nakata et al. (2022) proposed a k-Nearest Neighbor (KNN) classifier based on a zero-shot pretained feature extractor named CLIP (Radford et al., 2021), and it is evaluated and achieves state-of-arts on several popular benchmarks. EEIL (Castro et al., 2018) introduces the knowledge distillation loss in an end-to-end framework. NCM (Hou et al., 2019) employs three rectification components in the loss function to discriminate the old and new classes by magnitude and spatial orientation.

**Few-shot class incremental learning (FSCIL).** FSCIL is a recent active research area derived from CIL, aiming to classify inputs with few shot labeled samples.

Two major strategies are employed by recent FSCIL methods. The first one is knowledge representation learning and refinement. TOPIC (Tao et al., 2020) incorporates a neural gas network (Martinetz et al., 1991) to obtain and store the topology structure of the feature space. Zhang et al. (2021) proposed the Continually Evolved Classifier (CEC) which decoupled representation learning and classification and employed a graph model as the representation learner to propagate the global context between old and new sessions. Self-Promoted Prototype Refinement (SPPR)(Zhu et al., 2021) constructs a cosine-similarity-based relation matrix between old and new classes, which acts as a transitional coefficient for adapting old prototypes in the new feature space. The ALICE(Peng et al., 2022) framework, incorporates the angular penalty loss to adapt the feature extractor to obtain well-clustered features. Another strategy is knowledge distillation. Dong et al. (2021) proposed ERL++, an exemplar relation graph, to preserve the angular-based structural relations from old classes, and the distillation loss is used to retain the relation information in the new session.

**Semi-Supervised FSCIL (SSFSCIL).** SSFSCIL algorithms introduce additional unlabeled data to the FSCIL task to mitigate overfitting. From a representation learning perspective, FeSSSS (Ahmad et al., 2022) incorporates a self-learning ResNet50 encoder to a pretrained supervised ResNet18 model by a feature fusion design that concatenates the outputs of the two feature extractors. Kalla and Biswas (2022) proposed S3C which uses the stochastic classifier with a self-supervised feature extractor and its weights are kept frozen in the incremental session to reduce catastrophic forgetting. There are also some methods based on knowledge distillation. Cui et al. (2021) introduced a detailed semi-supervised configuration for a distillation-based network by revising the network until all unlabeled data obtain high-confidence pseudo-labels. Then unlabeled samples are combined with labeled data to enhance the performance of FSCIL. They further proposed Us-KD (Cui et al., 2022) which incorporates an uncertainty-guided component to filter out low-certainty unlabeled data to alleviate overfitting and noise during knowledge transfer. In their latest work (Cui et al., 2023), they observed that easily classifiable classes necessitate fewer unlabeled samples to achieve a high prediction accuracy, thus a data selection method was devised to avoid contaminating well-learned classes by less-reliable unlabeled data.

## 2.3 Incremental Learning with Probabilistic PCA

This section presents a method for semi-supervised incremental learning using probabilistic PCA (PPCA). First, an overview of the probabilistic PCA model is introduced, followed by k-means clustering using the

Mahalanobis distance. The Mahalanobis distance could be obtained from PPCA models or from standard Gaussian models. The proposed algorithm for semi-supervised incremental learning is introduced, as well as an efficient method for maintaining the cluster models based on running averages.

### 2.3.1 Probabilistic PCA Representation

Probabilistic PCA (PPCA) (Tipping and Bishop, 1999b) models a cluster of data with a Gaussian probability-cloud proxy in a lower-dimensional subspace with a mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The PPCA model will be used as the model for each class (cluster), but instead of classifying the data based on posterior and estimating the models by EM in Tipping and Bishop (1999a), our approach uses the Mahalanobis distance and performs estimation by k-Means.

PPCA is a plane with noise, where a $d$-dimensional observable variable $\mathbf{x}$ is approximated using a latent q-dimensional variable $\mathbf{t}$ as:

$$\mathbf{x} = \mathbf{W}\mathbf{t} + \boldsymbol{\mu} + \boldsymbol{\epsilon}, \tag{2.1}$$

where $\mathbf{t} \sim \mathcal{N}(0, \mathbf{I}_q)$, and $\boldsymbol{\mu} \in \mathbb{R}^d$ is the mean of observable variable $\mathbf{x}$. The transformation matrix $\mathbf{W} \in \mathbb{R}^{d \times q}$ connects the $d$-dimensional space and the lower $q$-dimensional subspace, and $\boldsymbol{\epsilon}$ represents the i.i.d noise following a Gaussian distribution with a diagonal covariance matrix $\mathcal{N}(0, \boldsymbol{\Psi})$. Then the conditional distribution of $\mathbf{x}$ given the latent variable $\mathbf{t}$ is

$$\mathbf{x}|\mathbf{t} = t \sim \mathcal{N}(\mathbf{W}t + \boldsymbol{\mu}, \boldsymbol{\Psi}). \tag{2.2}$$

The marginal distribution of $\mathbf{x}$ can be obtained by integration on the latent variable $\mathbf{t}$, obtaining $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = \mathbf{W}\mathbf{W}^T + \boldsymbol{\Psi}$.

To better capture the principal information of $\mathbf{x}$ in the lower dimensional subspace, Principal Component Analysis (PCA) (Pearson, 1901; Hotelling, 1933) is naturally considered in the factor analysis. If the singular value decomposition (SVD) is applied to the sample covariance matrix, we have

$$\mathbf{V}\mathbf{S}\mathbf{V}^T = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)^T \tag{2.3}$$

where $\mathbf{V}$ is the matrix containing the eigenvectors as columns, and $\mathbf{S}$ is the diagonal matrix with eigenvalues as diagonal elements.

Tipping and Bishop (1999a) show that PCA arises from factor analysis when $\boldsymbol{\Psi} = \sigma^2 \boldsymbol{I}$ and the d - q minor eigenvalues of the sample covariance matrix are equal. In this respect, the covariance matrix in $\boldsymbol{\Sigma}$

above can be defined as the combination of the reconstruction from the principal components which span the lower dimensional space, and the covariance from the noise term. It can be written as

$$\hat{\mathbf{\Sigma}} = \mathbf{L}^T\mathbf{D}\mathbf{L} + \lambda \mathbf{I}_d. \tag{2.4}$$

It is clear that $\mathbf{L}^T$ are formed by the first $q$ principal eigenvectors in $\mathbf{V}^T$, and $\mathbf{D}$ is the upper-left $q \times q$ sub-matrix of the eigenvalue matrix $\mathbf{S}$, which includes the first $q$ principal eigenvalues as diagonal elements. The parameter $\lambda > 0$ is a small number (e.g. $\lambda = 0.01$ in our experiments) indicating the variance $\sigma^2$ of the noise.

### 2.3.2 $k$-Means Clustering with the Mahalanobis Distance ($k$-PPCAs)

When using PPCA for classification, the data in each class is assumed to be mostly separable from the other classes in the feature space. Then the data can be naturally modeled by a mixture of Gaussian distributions with different parameters for each class. For example, the data belonging to the $k$-th class follows the Gaussian distribution with its mean $\boldsymbol{\mu}_k$ and covariance $\mathbf{\Sigma}_k$. Then the likelihood of observing data $\mathbf{x}$ in a given class $k$ is

$$p(\mathbf{x}|y\!=\!k) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_k|^{1/2}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)). \tag{2.5}$$

After taking the negative log-likelihood, the score function of $k$-th class can be calculated as follows, without the constant $(2\pi)^{d/2}$:

$$s_k(\mathbf{x}) = -2\log p(\mathbf{x}|y\!=\!k) = \log|\mathbf{\Sigma}_k| + (\mathbf{x}-\boldsymbol{\mu}_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k) \tag{2.6}$$

where a smaller score corresponds to a higher likelihood.

**Mahalanobis Distance.** The Mahalanobis distance is a well-known measure of distance first posted by P. C. Mahalanobis in 1936 (Mahalanobis, 2018). It takes the correlations of the variables into consideration and measures the distance between a point and the mean of a distribution by the unit of standard deviation of this distribution.

The Mahalanobis distance of a $d$-dimensional point $\mathbf{x} \in \mathbb{R}^d$ from a Gaussian distribution $Q = \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ is defined as

$$d_M(\mathbf{x}, Q) = \sqrt{(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}. \tag{2.7}$$

In the our proposed $k$-PPCAs, a simplified score function from Eq. (2.6) is

$$r(\mathbf{x}) = d_M^2(\mathbf{x}, Q) = (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \tag{2.8}$$

which is a corresponding representation of Mahalanobis distance in the following modified k-Means clustering.

Given PPCA parameters $\theta = (\boldsymbol{\mu}, \mathbf{L}, \mathbf{S})$ and denoting by $\mathbf{s} \in \mathbb{R}^q$ the vector containing the first $q$ elements of $\mathbf{S}$, the score (2.8) can be computed faster using the following

**Theorem 1.** *(Wang and Barbu, 2022) The score* (2.8) *can also be computed as:*

$$r(\mathbf{x}; \boldsymbol{\mu}, \mathbf{L}, \mathbf{S}) = \|\mathbf{x} - \boldsymbol{\mu}\|^2/\lambda - \|\mathbf{u}(\mathbf{x})\|^2/\lambda, \tag{2.9}$$

*where* $\mathbf{u}(\mathbf{x}) = \mathrm{diag}(\frac{\mathbf{s}}{\sqrt{\mathbf{s}^2 + \lambda \mathbf{I}_q}})\mathbf{L}^T(\mathbf{x} - \boldsymbol{\mu})$*, and the determinant as:*

$$\log |\boldsymbol{\Sigma}| = (d - q) \log \lambda + \sum_{i=1}^{q} \log(s_i^2 + \lambda). \tag{2.10}$$

**k-Means Clustering with Mahalanobis Distance.** The $k$-means clustering method is a popular unsupervised learning algorithm (Lloyd, 1982). It divides observations into $k$ clusters by assigning each point to its closest cluster centroid, and recalculating the centroids as the means of the clusters based on their assigned observations. In order to capture the shape information of the clusters in feature space, the Euclidean distance can be replaced by the Mahalanobis distance and the centroids are initialized from the labeled samples in each incremental session, as it is discussed in the following section.

For $n$ observations $\mathbf{x}_1, ..., \mathbf{x}_n$, the first step is to assign them to their closest cluster centroid respectively,

$$y_i = \underset{k}{\mathrm{argmin}}\, r(\mathbf{x}) = \underset{k}{\mathrm{argmin}} (\mathbf{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k), \tag{2.11}$$

and then update the cluster parameters ($\hat{\boldsymbol{\mu}}_k$,$\hat{\boldsymbol{\Sigma}}_k$) for each cluster,

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{\{i|y_i=k\}} \boldsymbol{x}_i}{n_k} \tag{2.12}$$

$$\hat{\boldsymbol{\Sigma}}_k = \frac{\sum_{\{i|y_i=k\}} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)^T}{n_k - 1} \tag{2.13}$$

where $n_k$ is the number of assigned observations in the $k$-th cluster. The above two steps are iteratively performed until the parameters have converged. When combining $k$-Means clustering with PPCA models, the above covariance matrix is decomposed by Eq. (2.3) and is remodeled by the first $q$ principal components and the covariance of the probabilistic term, and it is formulated as follows,

$$\hat{\boldsymbol{\Sigma}}_k = \boldsymbol{L}_k^T \boldsymbol{D}_k^{\mathbf{2}} \boldsymbol{L}_k + \lambda \mathbf{I}_d \tag{2.14}$$

### 2.3.3 Semi-Supervised Class-Incremental k-Means

The whole $k$-PPCAs algorithm is described in Alg. 1 and the flowcharts are shown in 2.2 and 2.3.

**Algorithm 1** $k$-PPCAs

```
Input: Models $\theta_k = (\boldsymbol{\mu}_k, \mathbf{L}_k, \mathbf{S}_k)$ for existing classes $k \in C_o$, new training dataset $D_{train} = \{D^{(l)}_{train}, D^{(u)}_{train}\}$ for current incremental session, with labeled set $D^{(l)}_{train}$, unlabeled set $D^{(u)}_{train}$, and new classes $C_n$.
Output: Updated PPCA models $\theta_k = (\boldsymbol{\mu}_k, \mathbf{L}_k, \mathbf{S}_k)$ for each new class $k \in C_n$.
for each new class $k \in C_n$ do
  Initialize $\theta_k = (\boldsymbol{\mu}_k, \mathbf{L}_k, \mathbf{S}_k)$ by PCA with $q$ principal vectors on the labeled data from class $k$ using Eq. (2.3)
end for
for j=1 to $n_{iter}$ do
  Initialize RAVE $R_k = (0, \mathbf{0}, \mathbf{0})$ for each new class $k \in C_n$
  for each new observation $\mathbf{x} \in D_{train}$ do
    if $\mathbf{x} \in D^{(l)}_{train}$ then
      Set $k = y$ as the observation label $y$
    else
      Compute scores $r_k = r(\mathbf{x}, \boldsymbol{\mu}_k, \mathbf{L}_k, \mathbf{S}_k)$ for all classes $k \in C_n \cup C_o$.
      Obtain label $k = \text{argmin}_k\, r_k$.
    end if
    if $k \in C_n$ and $r_k < \mu_{r_k} + \tau\sigma_{r_k}$ then
      add $\mathbf{x}$ to $R_k$ using Eq. (2.18)
    end if
  end for
  for $k \in C_n$ do
    Obtain $\boldsymbol{\mu}_k$ and $\mathbf{\Sigma}_k$ from $R_k$ using Eq. (2.17)
    Obtain $\mathbf{S}_k$ from $\mathbf{V}_k\mathbf{S}_k^2\mathbf{V}_k^T = \mathbf{\Sigma}_k$ using SVD
    Obtain $\mathbf{L}_k$ as the first $q$ columns of $\mathbf{V}_k$
  end for
end for
```

In a preprocessing step, a pretrained and frozen feature extractor is utilized to generate features from images or any other types of raw data. In this dissertation we used CLIP (Radford et al., 2021), a flexible and general contrastive learning method that builds two models, one from an image and one from an associated text, with maximum similarity between the image features and the text embedding features. The image encoder pre-trained by CLIP is introduced and kept frozen as the feature extractor for images in our experiments.

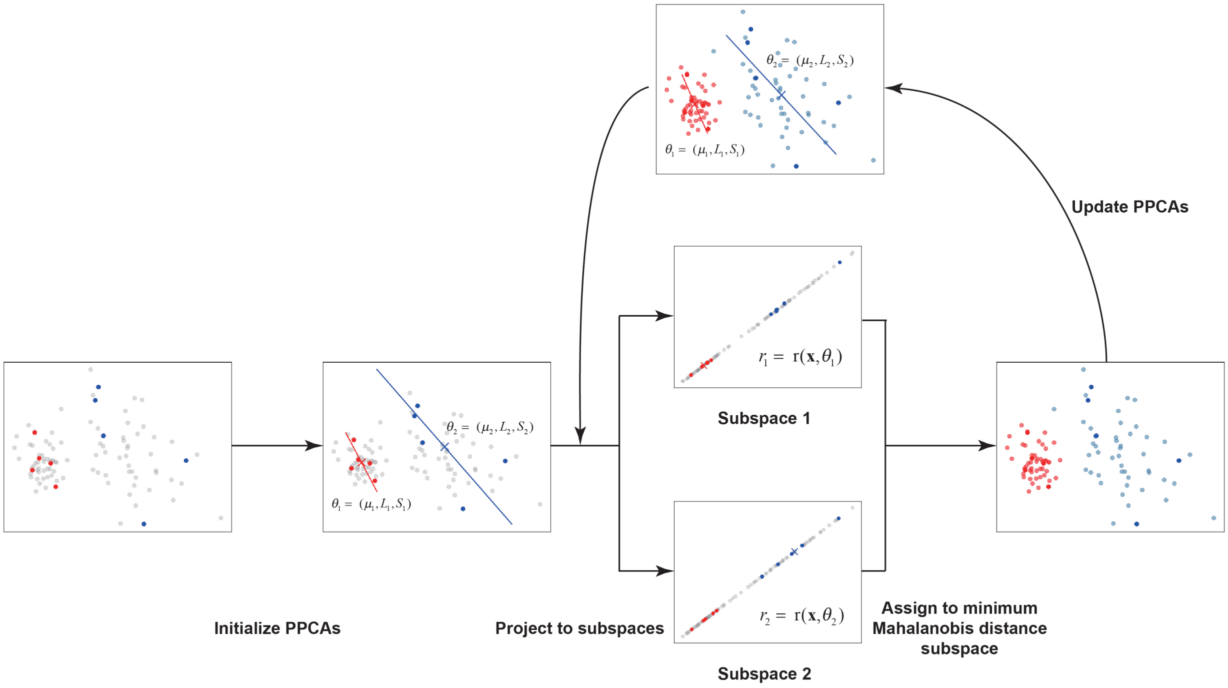

Figure 2.2: k-PPCAs in the base session under 2-way 5-shot setting with 2-Dimensional samples

For the base session and successive incremental sessions, the training dataset $D_{train}$ includes the subset with few-shot labeled features $D_{train}^{(l)}$ and the relatively massive subset of unlabeled features $D_{train}^{(u)}$. Then for each class in the new training set $D_{train}$, Principal Component Analysis (PCA) is applied to the covariance matrix of features labeled in that class, and the PPCA parameters $\theta = (\boldsymbol{\mu}, \mathbf{L}, \mathbf{S})$ are initialized from PCA, as described in Algorithm 2.

**Algorithm 2** Initialization with PCA

```
Input: Labeled dataset $D^{(l)} = \{D_1^{(l)}, D_2^{(l)}, ..., D_k^{(l)}, ..., D_K^{(l)}\}$ for all classes $k \in C$. $C$ is the set of classes
Output: Initialized PPCA models $\theta_k = (\boldsymbol{\mu}_k, \mathbf{L}_k, \mathbf{S}_k)$ for each class $k \in C$.
for each class $k \in C$ do
    Obtain $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ from $D_k^{(l)}$ using Eq. (2.12) and Eq. (2.14) respectively.
    Obtain $\mathbf{S}_k$ from $\mathbf{V}_k\mathbf{S}_k^2\mathbf{V}_k^T = \boldsymbol{\Sigma}_k$ using SVD
    Obtain $\mathbf{L}_k$ as the first $q$ columns of $\mathbf{V}_k$
end for
```

After initialization of PPCA models, for the base session $D_{train,0}$, the set of existing classes $C_o$ is empty, so all classes will be involved in $k$-means and the PPCA models will be updated for each class in the base session. The labeled observations will be directly assigned to the clusters corresponding to their labels. Then

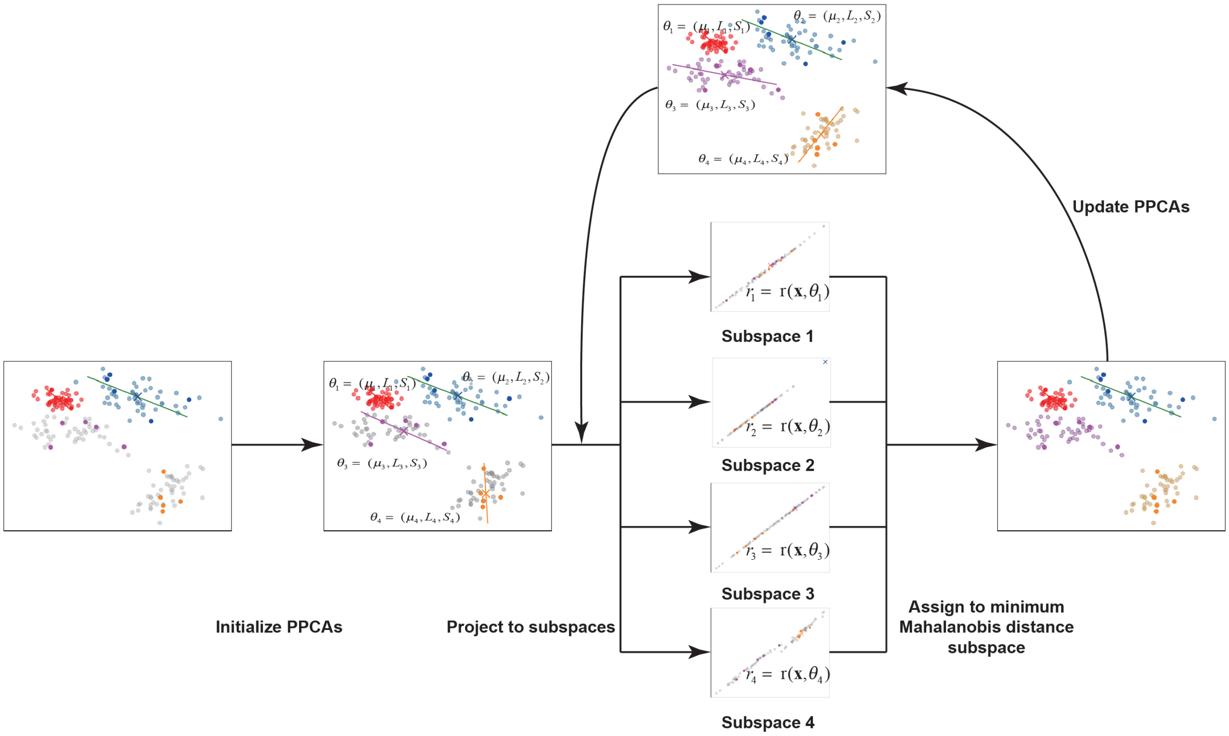
$\theta_1 = (\mu_1, L_1, S_1)$
$\theta_2 = (\mu_2, L_2, S_2)$
$\theta_3 = (\mu_3, L_3, S_3)$
$\theta_4 = (\mu_4, L_4, S_4)$
Update PPCAs
$r_1 = \mathrm{r}(\mathbf{x}, \theta_1)$
Subspace 1
$r_2 = \mathrm{r}(\mathbf{x}, \theta_2)$
Subspace 2
$r_3 = \mathrm{r}(\mathbf{x}, \theta_3)$
Subspace 3
$r_4 = \mathrm{r}(\mathbf{x}, \theta_4)$
Subspace 4
Initialize PPCAs
Project to subspaces
Assign to minimum Mahalanobis distance subspace

Figure 2.3: k-PPCAs in the incremental session under 2-way 5-shot setting with 2-Dimensional samples

for the unlabeled observations, the Mahalanobis distance between them and each cluster will be calculated as a score function using Eq. (2.9). Then each unlabeled observation will receive a pseudo-label that is the cluster id with the smallest score, and thus will be assigned to the corresponding cluster with the shortest Mahalanobis distance.

In order to improve the robustness of our method, the scores of observations within each cluster will be compared. Those observations with obviously large Mahalanobis distances will be identified as extreme values and removed from the model update step. In the experiment on the CUB200 dataset, the mean ($\mu_{r_k}$) and standard deviation ($\sigma_{r_k}$) of scores in each cluster are calculated, and each observation whose score is larger than $\mu_{r_k} + 1.96\sigma_{r_k}$ will be labeled as an extreme value. We use the criteria of $\mu_{r_k} + 1.96\sigma_{r_k}$ because under the assumption that the scores in each class follow a normal distribution, only approximately 2.5% of scores are expected to be larger than this criteria. After this filtering, the mean and the covariance will be updated for each cluster using Eq. (2.12) and Eq. (2.14) respectively. Then, by applying singular value decomposition (SVD) to updated covariance in each cluster, for example, for $k$-th PPCA model, the updated parameter $\mathbf{S}_k$ are obtained directly from the SVD, $\mathbf{V}_k\mathbf{S}_k^2\mathbf{V}_k^T = \mathbf{\Sigma}_k$, and the principal vectors $\mathbf{L}_k$

are obtained from the first $q$ columns of $\mathbf{V}_k$. After training the base session, the updated PPCA models will be stored and frozen for the following incremental sessions, and the classes in the base session will be listed in the set of existing classes $C_o$.

For the incremental sessions, the initialized PPCA models for the new classes will be appended to the inherited PPCA models for the existing classes. Then all PPCA models will be involved in the calculation of scores and the assignment of pseudo-labels for the incremental observations, but the PPCA models for the existing classes will be kept frozen and only the models for the new classes will be updated during training. After training an incremental session, the same procedures used in processing the base session are followed, which means that all models are stored for the next incremental session and the new classes will be treated as existing classes. The new PPCA models are continually added as new clusters until all incremental sessions are processed.

In practice, the lines 6-16 from Algorithm 1, which assign labels based on the current models and update the RAVEs, are done using mini-batches for efficient computation on the GPU. Mini-batches and running averages (Sun et al., 2024) (RAVE) are used to compute more efficiently the means and covariances for large datasets. RAVE is an online learning technique that can be employed to compute the mean and the covariance for each class incrementally. The details about how to compute the mean and the covariance from mini-batches will be introduced in the next subsection.

### 2.3.4 Maintaining Running Averages for Computing Covariance Matrices

Given a set of observations $\mathbf{x}_i, i \in J$, we need to compute $\hat{\boldsymbol{\mu}}_J = \frac{1}{|J|}\sum_{i \in J} \mathbf{x}_i$ and

$$\hat{\boldsymbol{\Sigma}}_J = \frac{\sum_{\{i \in J\}}(\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T}{|J| - 1}. \tag{2.15}$$

This can be done efficiently by maintaining running averages $R_J = (|J|, \mathbf{S}_x^{(J)}, \mathbf{S}_{xx}^{(J)})$, where:

$$\mathbf{S}_x^{(J)} = \frac{1}{|J|}\sum_{i \in J} \mathbf{x}_i, \mathbf{S}_{xx}^{(J)} = \frac{1}{|J|}\sum_{i \in J} \mathbf{x}_i \mathbf{x}_i^T. \tag{2.16}$$

From these running averages we can obtain $\hat{\boldsymbol{\mu}}_J = \mathbf{S}_x^{(J)}$ and

$$\hat{\boldsymbol{\Sigma}}_J = \frac{|J|}{|J| - 1}(\mathbf{S}_{xx}^{(J)} - \hat{\boldsymbol{\mu}}_J \hat{\boldsymbol{\mu}}_J^T) \tag{2.17}$$

The running averages can be updated incrementally, e.g. one observation at a time. For simplicity of notation, assume that $J = \{1, ..., n\}$ and denote $\mathbf{S}_x^{(J)} = \mathbf{S}_x^{(n)}$ and $\mathbf{S}_{xx}^{(J)} = \mathbf{S}_{xx}^{(n)}$. Then the running averages can be updated as follows:

$$\mathbf{S}_x^{(n)} = \frac{n-1}{n}\mathbf{S}_x^{(n-1)} + \frac{1}{n}\mathbf{x}_n \tag{2.18}$$

and similarly for $\mathbf{S}_{xx}^{(n)}$.

# CHAPTER 3

# EXPERIMENTAL EVALUATION AND DISCUSSION FOR INCREMENTAL K-PPCAS

## 3.1 Experiments

### 3.1.1 Datasets

Experiments are conducted on four popular datasets, the Caltech-UCSD Birds-200-2011 (CUB200), CIFAR100, miniImageNet and ILSVRC-2012 (ImageNet-1k) datasets. The first three datasets are widely used benchmark datasets for FSCIL, while ImageNet-1k is one of the most popular datasets for classification and incremental learning. Tao et al. (2020) provided a series of training schemes for few-shot class incremental learning on CUB200, CIFAR100, and miniImageNet. Their setting is widely adopted by the subsequent FSCIL research, e.g. Zhang et al. (2021), Ahmad et al. (2022). So for the convenience of comparison with previous related works, the experiments on these three datasets will follow the same schedules.

**Caltech-UCSD Birds-200-2011.** CUB200 (Wah et al., 2011) includes 5994 training and 5794 test fine-grained images about 200 classes of birds. In Tao et al. (2020), 100 classes were selected as the base session and the remaining 100 classes were equally divided into 10 incremental sessions. In our experiments, for each session, base and incremental, five samples per class are utilized with labels and the remaining samples are treated as unlabeled. This schedule follows a 10-way 5-shot setting.

**CIFAR100.** CIFAR100 (Krizhevsky et al., 2009) comprises 60000 small scale images equally distributed in 100 classes. For each class, there are 500 training images and 100 test images. In Tao et al. (2020), the 100 classes are separated into a base session with 60 classes and 5 classes for each of eight incremental sessions. The same 5-way 5-shot settings was used in this dissertation for CIFAR100.

**miniImageNet.** miniImageNet is a 100 class subset of ImageNet (Russakovsky et al., 2015), where we used the same 100 classes as Ravi and Larochelle (2016). Each class contains 500 training images and 100 test images. Similar to CIFAR100, Tao et al. (2020) splits the 100 classes into 60 classes for the base session and 40 classes for eight incremental sessions, thus 5 classes for each session. In our experiment, 5 samples are labeled in each of base session and incremental sessions, so a 5-way 5-shot setting is followed. In the ablation study, we find that applying data augmentation and preserving more principal components in PPCA can further improve the performance, and the accuracy is also improved when a stricter criteria for extreme

Table 3.1: Comparison of $k$-PPCAs with the state-of-the-art on the CUB200 dataset under 10-way 5-shot settings with 100 base and 100 incremental classes. † indicates results reported in Ahmad et al. (2022), * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. Our results are obtained from five independent runs with the standard deviation in parentheses.

| Method | Acc. in each session(%) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| NCM†(Hou et al., 2019) | 68.68 | 57.12 | 44.21 | 28.78 | 26.71 | 25.66 | 24.62 | 21.52 | 20.12 | 20.06 | 19.87 |
| EEIL†(Castro et al., 2018) | 68.68 | 53.63 | 47.91 | 44.2 | 36.3 | 27.46 | 25.93 | 24.7 | 23.95 | 24.13 | 22.11 |
| iCaRL†(Rebuffi et al., 2017) | 68.68 | 52.65 | 48.61 | 44.16 | 36.62 | 29.52 | 27.83 | 26.26 | 24.01 | 23.89 | 21.16 |
| TOPIC†(Tao et al., 2020) | 68.68 | 62.49 | 54.81 | 49.99 | 45.25 | 41.4 | 38.35 | 35.36 | 32.22 | 28.31 | 26.28 |
| LEC-Net†(Yang et al., 2021) | 70.86 | 58.15 | 54.83 | 49.34 | 45.85 | 40.55 | 39.70 | 34.59 | 36.58 | 33.56 | 31.96 |
| SS-iCaRL†(Cui et al., 2021) | 69.89 | 61.24 | 55.81 | 50.99 | 48.18 | 46.91 | 43.99 | 39.78 | 37.50 | 34.54 | 31.33 |
| SS-NCM†(Cui et al., 2021) | 69.89 | 61.91 | 55.51 | 51.71 | 49.68 | 46.11 | 42.19 | 39.03 | 37.96 | 34.05 | 32.65 |
| SPPR†(Zhu et al., 2021) | 68.68 | 61.85 | 57.43 | 52.68 | 50.19 | 46.88 | 44.65 | 43.07 | 40.17 | 39.63 | 37.33 |
| SS-NCM-CNN†(Cui et al., 2021) | 69.89 | 64.87 | 59.82 | 55.14 | 52.48 | 49.60 | 47.87 | 45.10 | 40.47 | 38.10 | 35.25 |
| Decoupled-DeepEMD†*(Zhang et al., 2020) | 75.35 | 70.69 | 66.68 | 62.34 | 59.76 | 56.54 | 54.61 | 52.52 | 50.73 | 49.20 | 47.60 |
| Decoupled-Cosine†*(Vinyals et al., 2016) | 75.52 | 70.95 | 66.46 | 61.20 | 60.86 | 56.88 | 55.40 | 53.49 | 51.94 | 50.93 | 49.31 |
| ERL†(Dong et al., 2021) | 73.52 | 70.12 | 65.12 | 62.01 | 58.56 | 57.99 | 56.77 | 56.52 | 55.01 | 53.68 | 50.01 |
| ERL++†(Dong et al., 2021) | 73.52 | 71.09 | 66.13 | 63.25 | 59.49 | 59.89 | 58.64 | 57.72 | 56.15 | 54.75 | 52.28 |
| CEC†(Zhang et al., 2021) | 75.85 | 71.94 | 68.50 | 63.5 | 62.43 | 58.27 | 57.73 | 55.81 | 54.83 | 53.52 | 52.28 |
| FeSSSS†(Ahmad et al., 2022) | 79.60 | 73.46 | 70.32 | 66.38 | 63.97 | 59.63 | 58.19 | 57.56 | 55.01 | 54.31 | 52.98 |
| $k$-Means | 69.27 | 67.38 | 66.37 | 64.10 | 63.56 | 62.23 | 62.48 | 62.03 | 61.00 | 61.21 | 61.64 |
| | (1.12) | (0.79) | (0.73) | (0.72) | (0.60) | (0.39) | (0.24) | (0.70) | (0.81) | (0.73) | (0.70) |
| $k$-PPCAs | 68.77 | 67.24 | 66.25 | 63.64 | 63.13 | 61.95 | 61.80 | 60.99 | 59.76 | 60.08 | 60.44 |
| | (1.23) | (1.32) | (1.10) | (1.16) | (1.33) | (1.32) | (1.27) | (1.37) | (1.45) | (1.42) | (1.42) |

values is used. The highest accuracy of miniImageNet is listed in the Tab. 3.3, while the other settings are detailed in Sec. 3.2.1.

**ImageNet.** ImageNet (Russakovsky et al., 2015) is one of the most popular datasets for image classification and incremental learning. Compared to the other datasets mentioned above, ImageNet is a relatively large dataset including 1,281,167 training and 50,000 validation images covering 1000 classes of objects. For each class, there are about 1,300 training images and 50 validation images. To the best of our knowledge, we are the first to report FSCIL results on ImageNet. In our experiment, the dataset is split into 500 classes for the base session and 100 new classes for five incremental sessions. In order to test the limit of our approach, we only provide 2 labeled samples for each of the base and incremental sessions, and the remaining images in the training set are treated as unlabeled samples. So we applied a 100-way 2-shot settings for this experiment.

Table 3.2: Experiments on CIFAR100 under 5-way 5-shot settings with 60 base and 40 incremental classes. † indicates results reported in Ahmad et al. (2022), * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. Our results are obtained from five independent runs, with the standard deviation in parentheses.

| Method | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| iCaRL†(Rebuffi et al., 2017) | 64.1 | 53.28 | 41.69 | 34.13 | 27.93 | 25.06 | 20.41 | 15.48 | 13.73 |
| EEIL†(Castro et al., 2018) | 64.1 | 53.11 | 43.71 | 35.15 | 28.96 | 24.98 | 21.01 | 17.26 | 15.85 |
| NCM†(Hou et al., 2019) | 64.1 | 53.05 | 43.96 | 36.97 | 31.61 | 26.73 | 21.23 | 16.78 | 13.54 |
| TOPIC†(Tao et al., 2020) | 64.1 | 55.88 | 47.07 | 45.16 | 40.11 | 36.38 | 33.96 | 31.55 | 29.37 |
| LEC-Net†(Yang et al., 2021) | 64.1 | 53.23 | 44.19 | 41.87 | 38.54 | 39.54 | 37.34 | 34.73 | 34.73 |
| SS-NCM-CNN†(Cui et al., 2021) | 64.1 | 62.22 | 61.11 | 58.0 | 54.22 | 50.66 | 48.88 | 46.0 | 44.44 |
| SPPR-ive†(Zhu et al., 2021) | 64.1 | 66.66 | 63.33 | 57.66 | 54.33 | 50.66 | 48.33 | 45.66 | 43.0 |
| Dec-DeepEMD†*(Zhang et al., 2020) | 69.75 | 65.06 | 61.2 | 57.21 | 53.88 | 51.40 | 48.80 | 46.84 | 44.41 |
| ERL†(Dong et al., 2021) | 73.62 | 66.79 | 63.67 | 60.54 | 56.98 | 53.63 | 50.92 | 48.73 | 46.33 |
| Dec.-Cosine†*(Vinyals et al., 2016) | 74.55 | 67.43 | 63.63 | 59.55 | 56.11 | 53.80 | 51.68 | 49.67 | 47.68 |
| ERL++†(Dong et al., 2021) | 73.62 | 68.22 | 65.14 | 61.84 | 58.35 | 55.54 | 52.51 | 50.16 | 48.23 |
| CEC†(Zhang et al., 2021) | 73.07 | 68.88 | 65.26 | 61.19 | 58.09 | 55.57 | 53.22 | 51.34 | 49.14 |
| SPPR†(Zhu et al., 2021) | 76.33 | 72.33 | 67.33 | 63.33 | 59.0 | 55.33 | 53.0 | 50.33 | 47.33 |
| FeSSSS†(Ahmad et al., 2022) | 75.35 | 70.81 | 66.7 | 62.73 | 59.62 | 56.45 | 54.33 | 52.10 | 50.23 |
| $k$-Means | 54.42 | 53.19 | 51.53 | 50.47 | 49.83 | 49.68 | 49.47 | 49.36 | 49.12 |
| | (1.10) | (1.03) | (1.31) | (1.14) | (0.87) | (0.83) | (0.52) | (0.51) | (0.54) |
| $k$-PPCAs | 64.53 | 64.01 | 63.28 | 61.62 | 61.14 | 61.08 | 61.55 | 61.38 | 60.98 |
| | (1.00) | (1.26) | (1.13) | (1.06) | (1.03) | (0.99) | (0.91) | (0.85) | (0.78) |

### 3.1.2 Implementation Details

All experiments are implemented with PyTorch and run on a RTX 3060 GPU. As mentioned in Sec. 2.3.3, the pre-trained image encoder of CLIP (Radford et al., 2021), which uses a ResNet-50x4 as its backbone, is adopted as feature extractor.

For image preprocessing on the CUB200, miniImageNet, and ImageNet datasets, we follow the settings recommended by the feature extractor, which is to resize the image so that the short edge is 288 while maintaining the original aspect ratio, followed by central cropping to obtain a 288×288 image as input for the feature extractor. For the CIFAR100 dataset, because of its low resolution 32×32 images, we tried two resizing scales with interpolation, 144×144 and 288×288 respectively. The images resized to 144×144 gave a better performance.

For all benchmarks, the model is trained for 10 epochs. The value of the PPCA parameter $\lambda$ in Eqns. (2.4) and (2.14) is set to $\lambda = 0.01$.

Table 3.3: Experiments on miniImageNet under 5-way 5-shot settings with 60 base and 40 incremental classes. † indicates results reported in Ahmad et al. (2022), * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. The results are obtained from five independent runs.

| Method | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| NCM†(Hou et al., 2019) | 61.31 | 47.8 | 39.31 | 31.91 | 25.68 | 21.35 | 18.67 | 17.24 | 14.17 |
| iCaRL†(Castro et al., 2018) | 61.31 | 46.32 | 42.94 | 37.63 | 30.49 | 24 | 20.89 | 18.8 | 17.21 |
| EEIL†(Castro et al., 2018) | 61.31 | 46.58 | 44 | 37.29 | 33.14 | 27.12 | 24.1 | 21.57 | 19.58 |
| LEC-Net†(Yang et al., 2021) | 61.31 | 35.37 | 36.66 | 38.59 | 33.90 | 35.89 | 36.12 | 32.97 | 30.55 |
| TOPIC†(Tao et al., 2020) | 61.31 | 50.09 | 45.17 | 41.16 | 37.48 | 35.52 | 32.19 | 29.46 | 24.42 |
| ERL†(Dong et al., 2021) | 61.67 | 56.19 | 54.70 | 51.19 | 47.61 | 45.23 | 44.0 | 40.95 | 39.8 |
| ERL++†(Dong et al., 2021) | 61.67 | 57.61 | 54.76 | 51.67 | 48.57 | 46.42 | 44.04 | 42.85 | 40.71 |
| SS-NCM-CNN†(Cui et al., 2021) | 62.88 | 60.88 | 57.63 | 52.8 | 50.66 | 48.28 | 45.27 | 41.65 | 41.21 |
| Dec-DeepEMD†*(Zhang et al., 2020) | 69.77 | 64.59 | 60.21 | 56.63 | 53.16 | 50.13 | 47.49 | 45.42 | 43.41 |
| Dec-Cosine†*(Vinyals et al., 2016) | 70.37 | 65.45 | 61.41 | 58.00 | 54.81 | 51.89 | 49.10 | 47.27 | 45.63 |
| CEC†(Zhang et al., 2021) | 72.00 | 66.83 | 62.97 | 59.43 | 56.70 | 53.73 | 51.19 | 49.24 | 47.63 |
| SPPR†(Zhu et al., 2021) | 80.0 | 74.0 | 68.66 | 64.33 | 61.0 | 57.33 | 54.66 | 51.66 | 49.0 |
| FeSSSS†(Ahmad et al., 2022) | 81.5 | 77.04 | 72.92 | 69.56 | 67.27 | 64.34 | 62.07 | 60.55 | 58.87 |
| $k$-Means | 62.46 | 62.95 | 61.96 | 62.22 | 62.49 | 62.34 | 61.84 | 62.15 | 62.37 |
| | (1.05) | (0.97) | (1.07) | (0.99) | (0.91) | (1.03) | (0.99) | (1.41) | (0.99) |
| $k$-PPCAs-Full Cov. | 62.46 | 62.95 | 61.96 | 62.24 | 62.51 | 62.34 | 61.85 | 62.15 | 62.51 |
| | (1.03) | (0.94) | (1.06) | (0.97) | (0.89) | (1.01) | (0.97) | (1.38) | (1.14) |
| $k$-PPCAs | 73.99 | 74.33 | 73.69 | 73.96 | 74.34 | 74.15 | 73.68 | 74.27 | 74.76 |
| | (1.16) | (1.07) | (1.05) | (0.96) | (0.93) | (0.86) | (0.85) | (0.82) | (0.76) |

**Data Augmentation.** Because the number of principal components $q$ is limited by the number of labeled samples per class, in order to overcome this constraint, data augmentation (random cropping and random horizontal flipping) was applied and 10 augmented images were generated for each labeled image. Based on the results from Tab. 3.5 in the ablation study, $q = 10$ principal components were used for all datasets. Five independent runs were conducted on each dataset and the mean and standard deviation of the accuracy are reported for each session.

### 3.1.3 Results

The results on CUB200, CIFAR100, miniImageNet and ImageNet datasets are reported in Tab. 3.1, Tab. 3.2, Tab. 3.3, and Tab. 3.4 respectively. Besides the average accuracy on the five independent runs, the standard deviation is also shown in parentheses. Tab. 3.1, Tab. 3.2, Tab. 3.3 present a comparison of $k$-PPCAs against other state-of-the-art FSCIL models (Tao et al., 2020; Yang et al., 2021; Cui et al., 2021; Zhu et al., 2021; Dong et al., 2021; Zhang et al., 2021; Ahmad et al., 2022). On these three datasets, CUB200,

Table 3.4: ImageNet-1k experiments under 100-way 2-shot settings with 500 base and 500 incremental classes. No other ImageNet results were found for FSCIL in the literature. The results are obtained from five independent runs.

| Method | Acc. in each session(%) with std. | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| $k$-Means | 53.68 | 54.99 | 54.03 | 52.81 | 51.49 | 51.93 |
| | (0.39) | (0.51) | (0.49) | (0.46) | (0.19) | (0.23) |
| $k$-PPCAs | 57.05 | 59.27 | 59.44 | 59.05 | 58.10 | 58.44 |
| | (0.27) | (0.33) | (0.52) | (0.44) | (0.32) | (0.27) |

Table 3.5: The influence of the number of principal components (PC) on accuracy for the miniImageNet dataset.

| Number | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| of PC | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 0 PC | 62.46 | 62.95 | 61.96 | 62.22 | 62.49 | 62.34 | 61.84 | 62.15 | 62.37 |
| $k$-Means | (1.05) | (0.97) | (1.07) | (0.99) | (0.91) | (1.03) | (0.99) | (1.41) | (0.99) |
| 5 PC | 73.19 | 73.47 | 72.82 | 73.05 | 73.44 | 73.30 | 72.91 | 73.39 | 73.92 |
| | (1.09) | (1.05) | (1.01) | (0.89) | (0.90) | (0.87) | (0.84) | (0.82) | (0.77) |
| 10 PC | 73.96 | 74.28 | 73.68 | 73.93 | 74.32 | 74.12 | 73.63 | 74.22 | 74.69 |
| | (1.29) | (1.17) | (1.13) | (1.08) | (1.03) | (0.95) | (0.93) | (0.90) | (0.83) |
| 20 PC | 74.00 | 74.24 | 73.44 | 73.71 | 74.20 | 73.87 | 73.27 | 73.89 | 74.42 |
| | (1.25) | (1.13) | (1.02) | (1.00) | (0.97) | (0.83) | (0.94) | (0.91) | (0.87) |
| 30 PC | 73.92 | 74.12 | 73.22 | 73.53 | 74.05 | 73.60 | 72.96 | 73.58 | 74.11 |
| | (1.31) | (1.17) | (1.15) | (1.13) | (1.11) | (0.91) | (0.89) | (0.85) | (0.82) |
| 50 PC | 73.87 | 74.07 | 73.10 | 72.97 | 73.45 | 72.87 | 72.04 | 72.76 | 73.31 |
| | (1.26) | (1.15) | (1.07) | (1.23) | (1.21) | (1.17) | (1.05) | (1.01) | (0.97) |
| All PC | 62.46 | 62.95 | 61.96 | 62.24 | 62.51 | 62.34 | 61.85 | 62.15 | 62.51 |
| Full Cov. | (1.03) | (0.94) | (1.06) | (0.97) | (0.89) | (1.01) | (0.97) | (1.38) | (1.14) |

CIFAR100, and miniImageNet, the proposed approach outperforms these models by a large margin. In Tab. 3.4 are shown the results on ImageNet-1k.

Tab. 3.1 shows the results on CUB200. Our approach performs better than other FSCIL approaches. Comparing the accuracy after the last incremental session, our approach obtains an accuracy of 60.44% and outperforms the second highest accuracy from FeSSSS (Ahmad et al., 2022) by 7.46%. On the CIFAR100 dataset, from Tab. 3.2, our approach reaches an accuracy of 60.98% and it exceeds the accuracy of FeSSSS by 10.75% after the eighth incremental session. On miniImageNet, shown in Tab. 3.3, our method obtains an accuracy of 74.76% and it outperforms the second highest approach, FeSSSS, by 15.89% after the last incremental session.

It is interesting to note that in the comparison on these three datasets, our method does not perform the best after the base session, but thanks to the robustness to catastrophic forgetting, our approach can retain a more stable accuracy and outperforms the other methods after updating on three or four incremental sessions.

As mentioned in Sec. 3.1.1, we may be the first to experiment an FSCIL method on the ImageNet-1k dataset. So only the result of our approach is provided in Tab. 3.4, and $k$-PPCAs achieve at an accuracy of $58.44\%$ on 1000 classes on ImageNet-1k under a 100-way 2-shot setting.

We also compared our method with a simple $k$-Means method adapted for the FSCIL task in a similar way as our $k$-PPCAs, but using the Euclidean distance as a score function. The results are documented as $k$-Means in Tab. 3.1, Tab. 3.2, Tab. 3.3, and Tab. 3.4. Our method outperforms the $k$-Means on CIFAR100, miniImageNet and ImageNet datasets. The $k$-Means method performs better than ours on CUB200 dataset by $1.2\%$ after the 10th incremental session, but considering the standard deviations of two results, we have observed that the difference between the two results is not statistically significant using a two-sample t-test ($p > 0.05$).

## 3.2 Discussion

### 3.2.1 Ablation Study

**The Number of Principal Components in PPCA.** In the proposed method, more principal components in PPCA can reconstruct more variation and shape information about the classes embedded in the feature space. But, when deciding the number of principal components, there is a trade-off because using more principal components may overfit the training data. As mentioned in Sec. 3.1.2, ten augmented images are generated for each labeled image under the 5-way 5-shot setting. Tab. 3.5 shows the miniImageNet accuracy of using different numbers of principal components, from 0 principal components, which is equivalent to a constant covariance in PPCA, i.e. $k$-Means, to containing all components, i.e. full covariance for each class. From the results, we can observe that the accuracy increases initially as more components are introduced in PPCAs, reaches the peak at the level of 10 principal components, then starts to drop if more principal components are added.

**The Criteria of Extreme Values.** As mentioned in Sec. 2.3.3, we assume the scores in each cluster follow a normal distribution, and during training any observation with a score larger than an extreme value criterion will be identified as extreme value and will be removed from the calculation of PPCA parameters. Experiments for extreme value criteria $\mu_{r_k} + \tau\sigma_{r_k}$ with different values of $\tau$ were conducted on miniImageNet, using $q = 10$ principal components. Tab. 3.6 shows that in general, a less strict criteria for extreme values

Table 3.6: The influence on accuracy of the threshold $\mu_{r_k} + \tau\sigma_{r_k}$ for removing extreme values for the miniImageNet dataset.

| | | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\tau$ | % | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 0 | 50 | 73.02 | 73.17 | 72.32 | 72.55 | 73.00 | 72.75 | 72.22 | 72.74 | 73.23 |
| | | (1.36) | (1.26) | (1.12) | (1.00) | (0.94) | (0.93) | (0.91) | (0.83) | (0.80) |
| 1 | 16 | 73.74 | 74.03 | 73.37 | 73.60 | 73.99 | 73.79 | 73.33 | 73.83 | 74.35 |
| | | (1.30) | (1.17) | (1.12) | (0.99) | (0.95) | (0.92) | (0.89) | (0.85) | (0.78) |
| 1.64 | 5 | 73.97 | 74.29 | 73.66 | 73.91 | 74.30 | 74.12 | 73.66 | 74.21 | 74.73 |
| | | (1.09) | (0.99) | (0.98) | (0.91) | (0.87) | (0.80) | (0.76) | (0.73) | (0.70) |
| 1.96 | 2.5 | 73.99 | 74.33 | 73.69 | 73.96 | 74.34 | 74.15 | 73.68 | 74.27 | 74.76 |
| | | (1.16) | (1.07) | (1.05) | (0.96) | (0.93) | (0.86) | (0.85) | (0.82) | (0.76) |
| 2.56 | 0.5 | 73.96 | 74.28 | 73.68 | 73.93 | 74.32 | 74.12 | 73.63 | 74.22 | 74.69 |
| | | (1.29) | (1.17) | (1.13) | (1.08) | (1.03) | (0.95) | (0.93) | (0.90) | (0.83) |
| $\infty$ | 0 | 73.97 | 74.31 | 73.70 | 73.97 | 74.33 | 74.14 | 73.67 | 74.25 | 74.72 |
| | | (1.11) | (1.03) | (0.98) | (0.91) | (0.89) | (0.83) | (0.82) | (0.79) | (0.74) |

contributes to a higher accuracy because more unlabeled features can help the estimation of PPCA parameters. But the accuracy will start to drop if a loose criteria (e.g. $\geq \mu_{r_k} + 2.56\sigma_{r_k}$ or all observations) cannot filter out the observations with a large Mahalanobis distance to their center.

### 3.2.2 Conclusion

Chapter Two and Three focus on the challenging yet practically significant task of Semi-Supervised Few-Shot Class Incremental Learning (SSFSCIL), where models are required to continually learn new classes from limited labeled samples and a relatively large pool of unlabeled data. We introduce $k$-PPCAs, an efficient probabilistic classifier built upon a pretrained self-supervised generic feature extractor. The $k$-PPCAs approach models classes using a mixture of probabilistic principal component analyzers, effectively capturing and distinguishing class variance in the feature space under the FSCIL setting.

Comprehensive experiments demonstrate the effectiveness of $k$-PPCAs on popular benchmarks like CUB200, CIFAR100, and miniImageNet. Additionally, we evaluate the method on the large-scale ImageNet-1k dataset, which had not previously been tested by SSFSCIL methods due to its size, showcasing the scalability of our approach.

The promising results on ImageNet-1k suggest that $k$-PPCAs can scale well to larger datasets, naturally extending the research to address challenges in large-scale few-shot learning.

Hierarchical clustering is widely used to uncover the hierarchical structure of clusters. There are two main types of approaches: agglomerative (bottom-up) and divisive (top-down). Agglomerative methods merge smaller clusters into larger ones to build a hierarchy, while divisive methods start with one large cluster and split it recursively. Human beings tend to organize categories of objects based on semantic hierarchies. Inspired by this, our next framework aims to integrate agglomerative clustering with the $k$-PPCAs method, creating a semi-supervised hierarchical classification system without requiring super-class annotations. This enhancement is expected to further improve classification performance, particularly on larger datasets with many more classes.

In the following Chapter Four, we explore how hierarchical clustering can reduce the computational complexity of large-scale few-shot learning, addressing the heavy computational burden associated with large output spaces.

## 3.3 Additional Experiments

### 3.3.1 Comparison with Other FSCIL Methods

In Sec. 3.1 are listed the results of comparing our methods against state-of-the-art SSFSCIL approaches. Tab. 3.7, Tab. 3.8, Tab. 3.9 displays the results of our method and other FSCIL methods, supervised and semi-supervised, on CUB200, CIFAR100, and miniImageNet respectively. FeSSSS(Ahmad et al., 2022), an SSFSCIL method, is still the second-best performer on all benchmarks. Specifically, for the other FSCIL methods, the samples in the base session are all provided as labeled.

### 3.3.2 Different Shots in the Base Session on CUB200 and CIFAR100

Considering the labeled data in the base session can also be expensive in reality, thus it is valuable to compare the few-shot base session and the conventional full-labeled design. We used a 5-shot base session and examined our method on all benchmarks. All experiments used CLIP-based ResNet50x4 as the feature extractor. Besides the result of miniImageNet provided in the main experiments, the results of CUB200 and CIFAR100 are listed in Tab. 3.10.

### 3.3.3 Comparison of $k$-Means and $k$-PPCAs on CUB200 and CIFAR100

Besides the comparison between $k$-PPCAs and the $k$-Means in Tab. 3.5 for ImageNet-1k, We also compared them for CUB200 and CIFAR100 based on a setting of the 5-shot base session. The results are documented in the Tab. 3.11. All experiments are performed with a CLIP-based ResNet50x4 as the backbone. The proposed method outperforms the $k$-Means on CIFAR100. On CUB200 the $k$-Means accuracy exceeds our proposed approach by 1.2%, but considering the standard deviations of two results, we have

Table 3.7: Comparison of $k$-PPCAs with the state-of-the-art FSCIL methods on the CUB200 dataset under 10-way 5-shot settings with 100 base and 100 incremental classes. † indicates results reported in Ahmad et al. (2022). * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. ‡ denotes the semi-supervised methods. The prefix CLIP- denotes the backbone pretrained by CLIP(Radford et al., 2021) and the backbones without a prefix are pretrained on ImageNet in a conventional way. Our results are obtained from five independent runs with the standard deviation in parentheses.

| Method | Acc. in each session(%) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| NCM†(Hou et al., 2019) | 68.68 | 57.12 | 44.21 | 28.78 | 26.71 | 25.66 | 24.62 | 21.52 | 20.12 | 20.06 | 19.87 |
| EEIL†(Castro et al., 2018) | 68.68 | 53.63 | 47.91 | 44.2 | 36.3 | 27.46 | 25.93 | 24.7 | 23.95 | 24.13 | 22.11 |
| iCaRL†(Rebuffi et al., 2017) | 68.68 | 52.65 | 48.61 | 44.16 | 36.62 | 29.52 | 27.83 | 26.26 | 24.01 | 23.89 | 21.16 |
| TOPIC†(Tao et al., 2020) | 68.68 | 62.49 | 54.81 | 49.99 | 45.25 | 41.4 | 38.35 | 35.36 | 32.22 | 28.31 | 26.28 |
| LEC-Net†(Yang et al., 2021) | 70.86 | 58.15 | 54.83 | 49.34 | 45.85 | 40.55 | 39.70 | 34.59 | 36.58 | 33.56 | 31.96 |
| SS-iCaRL†‡(Cui et al., 2021) | 69.89 | 61.24 | 55.81 | 50.99 | 48.18 | 46.91 | 43.99 | 39.78 | 37.50 | 34.54 | 31.33 |
| SS-NCM†‡(Cui et al., 2021) | 69.89 | 61.91 | 55.51 | 51.71 | 49.68 | 46.11 | 42.19 | 39.03 | 37.96 | 34.05 | 32.65 |
| SPPR†(Zhu et al., 2021) | 68.68 | 61.85 | 57.43 | 52.68 | 50.19 | 46.88 | 44.65 | 43.07 | 40.17 | 39.63 | 37.33 |
| SS-NCM-CNN†‡(Cui et al., 2021) | 69.89 | 64.87 | 59.82 | 55.14 | 52.48 | 49.60 | 47.87 | 45.10 | 40.47 | 38.10 | 35.25 |
| Decoupled-DeepEMD†*(Zhang et al., 2020) | 75.35 | 70.69 | 66.68 | 62.34 | 59.76 | 56.54 | 54.61 | 52.52 | 50.73 | 49.20 | 47.60 |
| Decoupled-Cosine†*(Vinyals et al., 2016) | 75.52 | 70.95 | 66.46 | 61.20 | 60.86 | 56.88 | 55.40 | 53.49 | 51.94 | 50.93 | 49.31 |
| ERL†(Dong et al., 2021) | 73.52 | 70.12 | 65.12 | 62.01 | 58.56 | 57.99 | 56.77 | 56.52 | 55.01 | 53.68 | 50.01 |
| ERL++†(Dong et al., 2021) | 73.52 | 71.09 | 66.13 | 63.25 | 59.49 | 59.89 | 58.64 | 57.72 | 56.15 | 54.75 | 52.28 |
| CEC†(Zhang et al., 2021) | 75.85 | 71.94 | 68.50 | 63.5 | 62.43 | 58.27 | 57.73 | 55.81 | 54.83 | 53.52 | 52.28 |
| FeSSSS†‡(Ahmad et al., 2022) | 79.60 | 73.46 | 70.32 | 66.38 | 63.97 | 59.63 | 58.19 | 57.56 | 55.01 | 54.31 | 52.98 |
| Us-KD‡(Cui et al., 2022) | 74.69 | 71.71 | 69.04 | 65.08 | 63.60 | 60.96 | 59.06 | 58.68 | 57.01 | 56.41 | 55.54 |
| S3C‡(Kalla and Biswas, 2022) | 80.62 | 77.55 | 73.19 | 68.54 | 68.05 | 64.33 | 63.58 | 62.07 | 60.61 | 59.79 | 58.95 |
| ALICE(Peng et al., 2022) | 77.4 | 72.7 | 70.6 | 67.2 | 65.9 | 63.4 | 62.9 | 61.9 | 60.5 | 60.6 | 60.1 |
| UaD-CE‡(Cui et al., 2023) | 75.17 | 73.27 | 70.87 | 67.14 | 65.49 | 63.66 | 62.42 | 62.55 | 60.99 | 60.48 | 60.72 |
| $k$-PPCAs | 81.46 | 78.84 | 76.87 | 73.09 | 71.90 | 70.02 | 69.24 | 67.93 | 66.25 | 66.18 | 66.06 |
| (CLIP-RN50x4) | (0.04) | (0.26) | (0.33) | (0.66) | (0.86) | (0.74) | (0.74) | (1.13) | (1.34) | (1.35) | (1.29) |
| $k$-PPCAs | 76.72 | 73.87 | 70.90 | 66.90 | 65.28 | 63.16 | 62.17 | 61.12 | 59.38 | 59.42 | 58.96 |
| (RN50) | (0.03) | (0.27) | (0.31) | (0.50) | (0.40) | (0.44) | (0.67) | (0.65) | (0.73) | (0.68) | (0.79) |
| $k$-PPCAs | 75.23 | 71.82 | 68.79 | 65.42 | 63.62 | 61.32 | 59.87 | 58.90 | 57.08 | 56.87 | 56.10 |
| (RN18) | (0.02) | (0.23) | (0.22) | (0.26) | (0.16) | (0.19) | (0.19) | (0.26) | (0.33) | (0.39) | (0.49) |

observed that the difference between the two results is not statistically significant using a two-sample t-test ($p > 0.05$).

### 3.3.4 Experiments on Various Backbones

The comparison of accuracy on various backbones is provided in Tab. 3.12. All experiments are performed with a setting of full-labeled base session and each labeled image is augmented to 10 images. The other hyperparameters have the same configuration as the approaches comparison in the main experiment. From the results, we can see that a larger backbone results in a higher performance, and the plain ResNet50 outperforms the CLIP-based model.

Table 3.8: Experiments on CIFAR100 under 5-way 5-shot settings with 60 base and 40 incremental classes. † indicates results reported in Ahmad et al. (2022). * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. ‡ denotes the semi-supervised methods. Only the accuracy of the end incremental session is provided in S3C(Kalla and Biswas, 2022). The prefix CLIP- denotes the backbone pretrained by CLIP(Radford et al., 2021) and the backbones without a prefix are pretrained on ImageNet in a conventional way. Our results are obtained from five independent runs.

| Method | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| iCaRL†(Rebuffi et al., 2017) | 64.1 | 53.28 | 41.69 | 34.13 | 27.93 | 25.06 | 20.41 | 15.48 | 13.73 |
| EEIL†(Castro et al., 2018) | 64.1 | 53.11 | 43.71 | 35.15 | 28.96 | 24.98 | 21.01 | 17.26 | 15.85 |
| NCM†(Hou et al., 2019) | 64.1 | 53.05 | 43.96 | 36.97 | 31.61 | 26.73 | 21.23 | 16.78 | 13.54 |
| TOPIC†(Tao et al., 2020) | 64.1 | 55.88 | 47.07 | 45.16 | 40.11 | 36.38 | 33.96 | 31.55 | 29.37 |
| LEC-Net†(Yang et al., 2021) | 64.1 | 53.23 | 44.19 | 41.87 | 38.54 | 39.54 | 37.34 | 34.73 | 34.73 |
| SS-NCM-CNN†‡(Cui et al., 2021) | 64.1 | 62.22 | 61.11 | 58.0 | 54.22 | 50.66 | 48.88 | 46.0 | 44.44 |
| SPPR-ive†(Zhu et al., 2021) | 64.1 | 66.66 | 63.33 | 57.66 | 54.33 | 50.66 | 48.33 | 45.66 | 43.0 |
| Dec-DeepEMD†*(Zhang et al., 2020) | 69.75 | 65.06 | 61.2 | 57.21 | 53.88 | 51.40 | 48.80 | 46.84 | 44.41 |
| ERL†(Dong et al., 2021) | 73.62 | 66.79 | 63.67 | 60.54 | 56.98 | 53.63 | 50.92 | 48.73 | 46.33 |
| Dec.-Cosine†*(Vinyals et al., 2016) | 74.55 | 67.43 | 63.63 | 59.55 | 56.11 | 53.80 | 51.68 | 49.67 | 47.68 |
| ERL++†(Dong et al., 2021) | 73.62 | 68.22 | 65.14 | 61.84 | 58.35 | 55.54 | 52.51 | 50.16 | 48.23 |
| CEC†(Zhang et al., 2021) | 73.07 | 68.88 | 65.26 | 61.19 | 58.09 | 55.57 | 53.22 | 51.34 | 49.14 |
| SPPR†(Zhu et al., 2021) | 76.33 | 72.33 | 67.33 | 63.33 | 59.0 | 55.33 | 53.0 | 50.33 | 47.33 |
| FeSSSS†‡(Ahmad et al., 2022) | 75.35 | 70.81 | 66.7 | 62.73 | 59.62 | 56.45 | 54.33 | 52.10 | 50.23 |
| S3C‡(Kalla and Biswas, 2022) | - | - | - | - | - | - | - | - | 53.96 |
| ALICE(Peng et al., 2022) | 79.0 | 70.5 | 67.1 | 63.4 | 61.2 | 59.2 | 58.1 | 56.3 | 54.1 |
| Us-KD‡(Cui et al., 2022) | 76.85 | 69.87 | 65.46 | 62.36 | 59.86 | 57.29 | 55.22 | 54.91 | 54.42 |
| UaD-CE‡(Cui et al., 2023) | 75.55 | 72.17 | 68.57 | 65.35 | 62.80 | 60.27 | 59.12 | 57.05 | 54.5 |
| $k$-PPCAs | 73.06 | 71.54 | 70.30 | 68.23 | 67.41 | 66.99 | 67.18 | 66.81 | 66.15 |
| (CLIP-RN50x4) | (0.07) | (0.15) | (0.24) | (0.26) | (0.17) | (0.18) | (0.15) | (0.18) | (0.15) |
| $k$-PPCAs | 73.29 | 72.26 | 71.16 | 68.53 | 68.41 | 67.59 | 67.53 | 66.76 | 65.34 |
| (RN50) | (0.06) | (0.09) | (0.32) | (0.32) | (0.36) | (0.26) | (0.21) | (0.16) | (0.19) |
| $k$-PPCAs | 69.87 | 68.83 | 67.83 | 65.39 | 65.05 | 63.73 | 63.74 | 62.88 | 61.40 |
| (RN18) | (0.07) | (0.10) | (0.33) | (0.34) | (0.34) | (0.32) | (0.32) | (0.31) | (0.28) |

Table 3.9: Experiments on miniImageNet under 5-way 5-shot settings with 60 base and 40 incremental classes. † indicates results reported in Ahmad et al. (2022). * denotes the few-shot approaches adapted by Zhang et al. (2021) for FSCIL. ‡ denotes the semi-supervised methods. Only the accuracy of the end incremental session is provided in S3C(Kalla and Biswas, 2022). The prefix CLIP- denotes the backbone pretrained by CLIP(Radford et al., 2021). The results are obtained from five independent runs.

| Method | Acc. in each session(%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| NCM†(Hou et al., 2019) | 61.31 | 47.8 | 39.31 | 31.91 | 25.68 | 21.35 | 18.67 | 17.24 | 14.17 |
| iCaRL†(Castro et al., 2018) | 61.31 | 46.32 | 42.94 | 37.63 | 30.49 | 24 | 20.89 | 18.8 | 17.21 |
| EEIL†(Castro et al., 2018) | 61.31 | 46.58 | 44 | 37.29 | 33.14 | 27.12 | 24.1 | 21.57 | 19.58 |
| LEC-Net†(Yang et al., 2021) | 61.31 | 35.37 | 36.66 | 38.59 | 33.90 | 35.89 | 36.12 | 32.97 | 30.55 |
| TOPIC†(Tao et al., 2020) | 61.31 | 50.09 | 45.17 | 41.16 | 37.48 | 35.52 | 32.19 | 29.46 | 24.42 |
| ERL†(Dong et al., 2021) | 61.67 | 56.19 | 54.70 | 51.19 | 47.61 | 45.23 | 44.0 | 40.95 | 39.8 |
| ERL++†(Dong et al., 2021) | 61.67 | 57.61 | 54.76 | 51.67 | 48.57 | 46.42 | 44.04 | 42.85 | 40.71 |
| SS-NCM-CNN†‡(Cui et al., 2021) | 62.88 | 60.88 | 57.63 | 52.8 | 50.66 | 48.28 | 45.27 | 41.65 | 41.21 |
| Dec-DeepEMD†*(Zhang et al., 2020) | 69.77 | 64.59 | 60.21 | 56.63 | 53.16 | 50.13 | 47.49 | 45.42 | 43.41 |
| Dec-Cosine†*(Vinyals et al., 2016) | 70.37 | 65.45 | 61.41 | 58.00 | 54.81 | 51.89 | 49.10 | 47.27 | 45.63 |
| CEC†(Zhang et al., 2021) | 72.00 | 66.83 | 62.97 | 59.43 | 56.70 | 53.73 | 51.19 | 49.24 | 47.63 |
| SPPR†(Zhu et al., 2021) | 80.0 | 74.0 | 68.66 | 64.33 | 61.0 | 57.33 | 54.66 | 51.66 | 49.0 |
| Us-KD‡(Cui et al., 2022) | 72.35 | 67.22 | 62.41 | 59.85 | 57.81 | 55.52 | 52.64 | 50.86 | 50.47 |
| UaD-CE‡(Cui et al., 2023) | 72.35 | 66.91 | 62.13 | 59.89 | 57.41 | 55.52 | 53.26 | 51.46 | 50.52 |
| S3C‡(Kalla and Biswas, 2022) | - | - | - | - | - | - | - | - | 52.14 |
| ALICE(Peng et al., 2022) | 80.6 | 70.6 | 67.4 | 64.5 | 62.5 | 60.0 | 57.8 | 56.8 | 55.7 |
| FeSSSS†‡(Ahmad et al., 2022) | 81.5 | 77.04 | 72.92 | 69.56 | 67.27 | 64.34 | 62.07 | 60.55 | 58.87 |
| $k$-PPCAs | 81.57 | 81.28 | 80.12 | 79.98 | 79.97 | 79.44 | 78.60 | 78.92 | 79.02 |
| (CLIP-RN50x4) | (0.05) | (0.04) | (0.11) | (0.11) | (0.11) | (0.07) | (0.08) | (0.07) | (0.06) |
| $k$-PPCAs | 76.82 | 76.17 | 75.06 | 75.03 | 75.01 | 74.61 | 73.75 | 73.97 | 74.19 |
| (CLIP-RN50) | (0.07) | (0.14) | (0.22) | (0.21) | (0.22) | (0.23) | (0.20) | (0.18) | (0.17) |

Table 3.10: Comparison of few-shot and all-labeled base session on CUB200 and CIFAR100. All experiments are performed with a CLIP-based ResNet50x4 as the backbone

| Benchmark | Base Session | Acc. in each session(%) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| CUB200 | 5-shot | 68.77 | 67.24 | 66.25 | 63.64 | 63.13 | 61.95 | 61.80 | 60.99 | 59.76 | 60.08 | 60.44 |
| | | (1.23) | (1.32) | (1.10) | (1.16) | (1.33) | (1.32) | (1.27) | (1.37) | (1.45) | (1.42) | (1.42) |
| | all-labeled | 81.46 | 78.84 | 76.87 | 73.09 | 71.90 | 70.02 | 69.24 | 67.93 | 66.25 | 66.18 | 66.06 |
| | | (0.04) | (0.26) | (0.33) | (0.66) | (0.86) | (0.74) | (0.74) | (1.13) | (1.34) | (1.35) | (1.29) |
| CIFAR100 | 5-shot | 64.53 | 64.01 | 63.28 | 61.62 | 61.14 | 61.08 | 61.55 | 61.38 | 60.98 | - | - |
| | | (1.00) | (1.26) | (1.13) | (1.06) | (1.03) | (0.99) | (0.91) | (0.85) | (0.78) | - | - |
| | all-labeled | 73.06 | 71.54 | 70.30 | 68.23 | 67.41 | 66.99 | 67.18 | 66.81 | 66.15 | - | - |
| | | (0.07) | (0.15) | (0.24) | (0.26) | (0.17) | (0.18) | (0.15) | (0.18) | (0.15) | - | - |

Table 3.11: Comparison of $k$-Means and $k$-PPCAs on CUB200 and CIFAR100. All experiments are performed with a CLIP-based ResNet50x4 as the backbone.

| Benchmark | Method | Acc. in each session(%) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| CUB200 | $k$-Means | 69.27 | 67.38 | 66.37 | 64.10 | 63.56 | 62.23 | 62.48 | 62.03 | 61.00 | 61.21 | 61.64 |
| | | (1.12) | (0.79) | (0.73) | (0.72) | (0.60) | (0.39) | (0.24) | (0.70) | (0.81) | (0.73) | (0.70) |
| | $k$-PPCAs | 68.77 | 67.24 | 66.25 | 63.64 | 63.13 | 61.95 | 61.80 | 60.99 | 59.76 | 60.08 | 60.44 |
| | | (1.23) | (1.32) | (1.10) | (1.16) | (1.33) | (1.32) | (1.27) | (1.37) | (1.45) | (1.42) | (1.42) |
| CIFAR100 | $k$-Means | 54.42 | 53.19 | 51.53 | 50.47 | 49.83 | 49.68 | 49.47 | 49.36 | 49.12 | - | - |
| | | (1.10) | (1.03) | (1.31) | (1.14) | (0.87) | (0.83) | (0.52) | (0.51) | (0.54) | - | - |
| | $k$-PPCAs | 64.53 | 64.01 | 63.28 | 61.62 | 61.14 | 61.08 | 61.55 | 61.38 | 60.98 | - | - |
| | | (1.00) | (1.26) | (1.13) | (1.06) | (1.03) | (0.99) | (0.91) | (0.85) | (0.78) | - | - |

Table 3.12: Comparison of various backbones on benchmarks. The prefix CLIP- denotes the backbone pretrained by CLIP(Radford et al., 2021) and the backbones without prefix are pretrained on ImageNet in a conventional way. A suffix -ft denotes the backbone is fine-tuned on the base session.

| Benchmark | Backbones | Acc. in each session(%) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| CUB200 | CLIP-RN50x4 | 81.46 | 78.84 | 76.87 | 73.09 | 71.90 | 70.02 | 69.24 | 67.93 | 66.25 | 66.18 | 66.06 |
| | | (0.04) | (0.26) | (0.33) | (0.66) | (0.86) | (0.74) | (0.74) | (1.13) | (1.34) | (1.35) | (1.29) |
| | CLIP-RN50 | 74.15 | 71.05 | 68.42 | 64.44 | 62.95 | 60.98 | 59.74 | 58.17 | 56.19 | 56.37 | 55.85 |
| | | (0.03) | (0.35) | (0.35) | (0.48) | (0.51) | (0.40) | (0.46) | (0.86) | (0.69) | (0.38) | (0.60) |
| | RN50 | 72.28 | 69.23 | 66.34 | 62.59 | 60.41 | 58.41 | 56.87 | 55.19 | 53.11 | 53.10 | 52.94 |
| | | (0.09) | (0.05) | (0.19) | (0.46) | (0.57) | (0.48) | (0.44) | (0.56) | (0.61) | (0.56) | (0.77) |
| | RN50-ft | 76.72 | 73.87 | 70.90 | 66.90 | 65.28 | 63.16 | 62.17 | 61.12 | 59.38 | 59.42 | 58.96 |
| | | (0.03) | (0.27) | (0.31) | (0.50) | (0.40) | (0.44) | (0.67) | (0.65) | (0.73) | (0.68) | (0.79) |
| | RN18 | 69.41 | 65.76 | 62.41 | 59.31 | 57.16 | 54.84 | 53.19 | 51.74 | 50.06 | 49.89 | 49.35 |
| | | (0.04) | (0.15) | (0.29) | (0.42) | (0.42) | (0.43) | (0.32) | (0.41) | (0.53) | (0.55) | (0.60) |
| | RN18-ft | 75.23 | 71.82 | 68.79 | 65.42 | 63.62 | 61.32 | 59.87 | 58.90 | 57.08 | 56.87 | 56.10 |
| | | (0.02) | (0.23) | (0.22) | (0.26) | (0.16) | (0.19) | (0.19) | (0.26) | (0.33) | (0.39) | (0.49) |
| CIFAR100 | CLIP-RN50x4 | 73.06 | 71.54 | 70.30 | 68.23 | 67.41 | 66.99 | 67.18 | 66.81 | 66.15 | - | - |
| | | (0.07) | (0.15) | (0.24) | (0.26) | (0.17) | (0.18) | (0.15) | (0.18) | (0.15) | - | - |
| | CLIP-RN50 | 66.36 | 64.24 | 63.05 | 61.07 | 60.40 | 59.60 | 59.47 | 58.48 | 57.56 | - | - |
| | | (0.11) | (0.22) | (0.30) | (0.34) | (0.19) | (0.16) | (0.13) | (0.19) | (0.30) | - | - |
| | RN50 | 73.29 | 72.26 | 71.16 | 68.53 | 68.41 | 67.59 | 67.53 | 66.76 | 65.34 | - | - |
| | | (0.06) | (0.09) | (0.32) | (0.32) | (0.36) | (0.26) | (0.21) | (0.16) | (0.19) | - | - |
| | RN18 | 69.87 | 68.83 | 67.83 | 65.39 | 65.05 | 63.73 | 63.74 | 62.88 | 61.40 | - | - |
| | | (0.07) | (0.10) | (0.33) | (0.34) | (0.34) | (0.32) | (0.32) | (0.31) | (0.28) | - | - |
| miniImageNet | CLIP-RN50x4 | 81.57 | 81.28 | 80.12 | 79.98 | 79.97 | 79.44 | 78.60 | 78.92 | 79.02 | - | - |
| | | (0.05) | (0.04) | (0.11) | (0.11) | (0.11) | (0.07) | (0.08) | (0.07) | (0.06) | - | - |
| | CLIP-RN50 | 76.82 | 76.17 | 75.06 | 75.03 | 75.01 | 74.61 | 73.75 | 73.97 | 74.19 | - | - |
| | | (0.07) | (0.14) | (0.22) | (0.21) | (0.22) | (0.23) | (0.20) | (0.18) | (0.17) | - | - |

# CHAPTER 4

# LARGE-SCALE FEW-SHOT CLASSIFICATION WITH SEMI-SUPERVISED HIERARCHICAL K-PROBABILISTIC PCAS

## 4.1 Introduction

In the past decades, the field of deep learning has witnessed remarkable advancements in various tasks Russakovsky et al. (2015); Lin et al. (2014); Everingham et al. (2010). Nonetheless, its efficacy is commonly contingent upon a fully-supervised context, necessitating extensive data and annotations. In practical situations, acquiring high-quality data annotations proves challenging, given its general expensiveness, time-intensive nature, and occasional reliance on domain knowledge. In contrast to labeled data, unlabeled data sourced from real-world scenarios is more cost-effective and readily available. Consequently, it is logical to harness their impact to enhance the performance of the model. Due to the effectiveness in handling limited labeled data and harnessing extensive unlabeled data, semi-supervised learning has gained considerable interest in the research community Grandvalet and Bengio (2004); Zhou and Li (2010). This approach notably diminishes the reliance on human labor, addressing the challenges associated with data annotation.

Few-shot learning, as another approach to address the high cost of data annotation, has gained widespread attention in recent yearsVinyals et al. (2016); Snell et al. (2017); Zhang et al. (2020); Ravi and Larochelle (2016); Finn et al. (2017); Qi et al. (2018); Chen et al. (2019). A notable hallmark for humans is the ability to rapidly establish cognitive capabilities for new concepts with just one or a few examples. Inspired by human learning methods, the goal of few-shot learning is to establish a classifier based on features extracted from very limited labeled data per class and amplify it with generalization for unknown observations.

A classic paradigm in few-shot learning is to train a classification model on a source dataset with a large amount of data and then fine-tune it on a target dataset with limited data. However, this approach may lead to model overfitting since a small amount of data may not reflect the large dataset's true distribution. To address this overfitting issue, few-shot learning methods based on auxiliary information, e.g. unlabeled data, have been proposed, which is the Semi-Supervised Few-Shot Learning task (SSFSL).

The prevailing fashion in SSFSL is to predict unlabeled data skillfully using pseudo-labels by carefully crafted strategies. Subsequently, this approach aims to enhance the exceptionally limited support set of

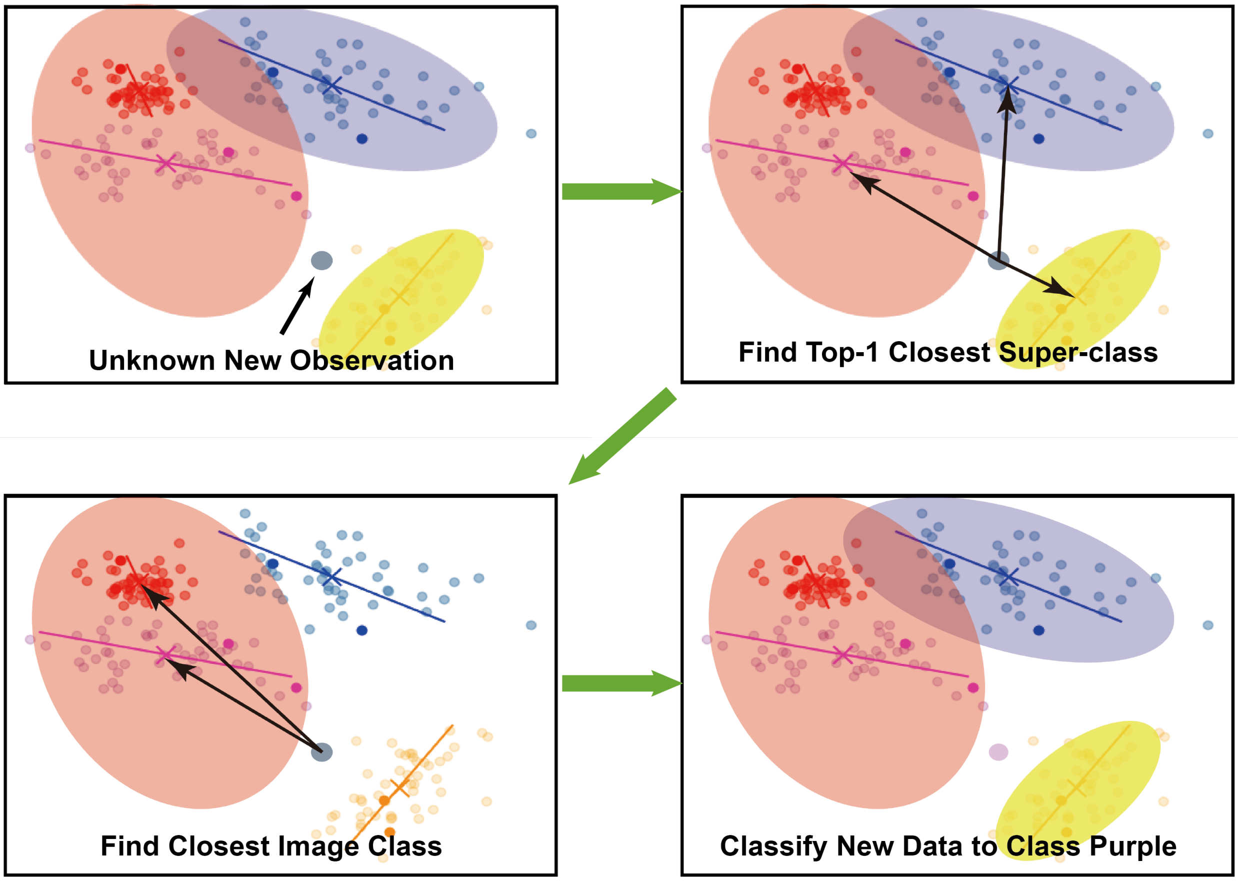
Unknown New Observation
Find Top-1 Closest Super-class
Find Closest Image Class
Classify New Data to Class Purple

Figure 4.1: Flowchart of H$k$-PPCAs Classification: Four image classes and three super-classes, number of candidate super-classes T=1. (The colored points are labeled or classified observations. The grey points are unlabeled.)

labeled data in few-shot classification. While there have been numerous works on SSFSL in recent years Ren et al. (2018); Liu et al. (2018); Wang et al. (2020); Lazarou et al. (2021); Huang et al. (2021), there are few results evaluated on large-scale multi-class datasets. Therefore, we follow the above fashion and propose a hierarchical SSFSL approach named H$k$-PPCAs designed for addressing challenges on large-scale datasets.

Our proposed method employs a two-level taxonomy to represent the semantic hierarchy of image classes, although additional levels can be incorporated as needed. Image classes are conceptualized as Gaussian distributions reconstructed using Probabilistic Principal Component Analyzers (PPCAs). The initialization involves using labeled observations to initialize image classes through PPCAs, followed by predicting pseudo-labels for unlabeled data based on the nearest image classes determined by Mahalanobis distance. The image classes are then iteratively updated until convergence. For the higher semantic level, image classes sharing similar semantic information are assumed to cluster together, forming semantic clusters characterized by Gaussians, referred to as super-classes. To uncover underlying semantic relations between image classes and establish the class hierarchy, a modified k-means clustering algorithm is introduced. After the construction of the hierarchical structure, an image during classification, such as a panda, will be initially assigned to a super-class like 'Ursidae' and be subsequently classified among the image classes associated with the 'Ursidae' super-class. The hierarchical design reduces the search space and demands significantly fewer computational resources compared to the flat structure. With a dataset containing K classes, the hierarchical model can classify images in $O(\sqrt{K})$ time, a notable improvement from the $O(K)$ time in the non-hierarchical classification. Experiments have demonstrated the effectiveness in both accuracy and efficiency of the proposed approach when applied to large-scale datasets.

The main contributions of the proposed method are:

- It proposes a semi-supervised hierarchical framework named H$k$-PPCAs for few-shot classification on large-scale datasets with thousands of classes. The hierarchical design formed by two-layer PPCAs reduces the classification time complexity to $O(\sqrt{K})$, and potentially to $O(\log K)$ if incorporating more layers.

- It introduces a training procedure utilizing a modified k-means clustering that groups class prototypes, which can classify the out-of-distribution samples as ”unknown” through a low-confidence score metric.

- Conducts experiments on the ImageNet-1k and ImageNet-10k datasets, which have not been used to evaluate FSL methods because of their large size.

## 4.2 Related Work

**Few-shot Learning.** Few-shot learning is a learning task that focuses on training models to make accurate predictions or classifications with very limited labeled examples. The term ”few-shot” typically refers to having a small number of labeled observations per class. Approaches to few-shot learning include meta-learning methods, which involve training models to quickly adapt to new tasks based on a limited number of examples, and transfer learning methods, which leverage knowledge learned from related tasks or domains to improve performance on new tasks with limited data.

Meta-learning-based methods, often referred to as ”learning-to-learn,” have two popular learning paradigms: metric-based methods (Vinyals et al., 2016; Snell et al., 2017; Zhang et al., 2020) and optimization-based methods (Ravi and Larochelle, 2016; Finn et al., 2017). Specifically, Prototypical Networks (Snell et al.,

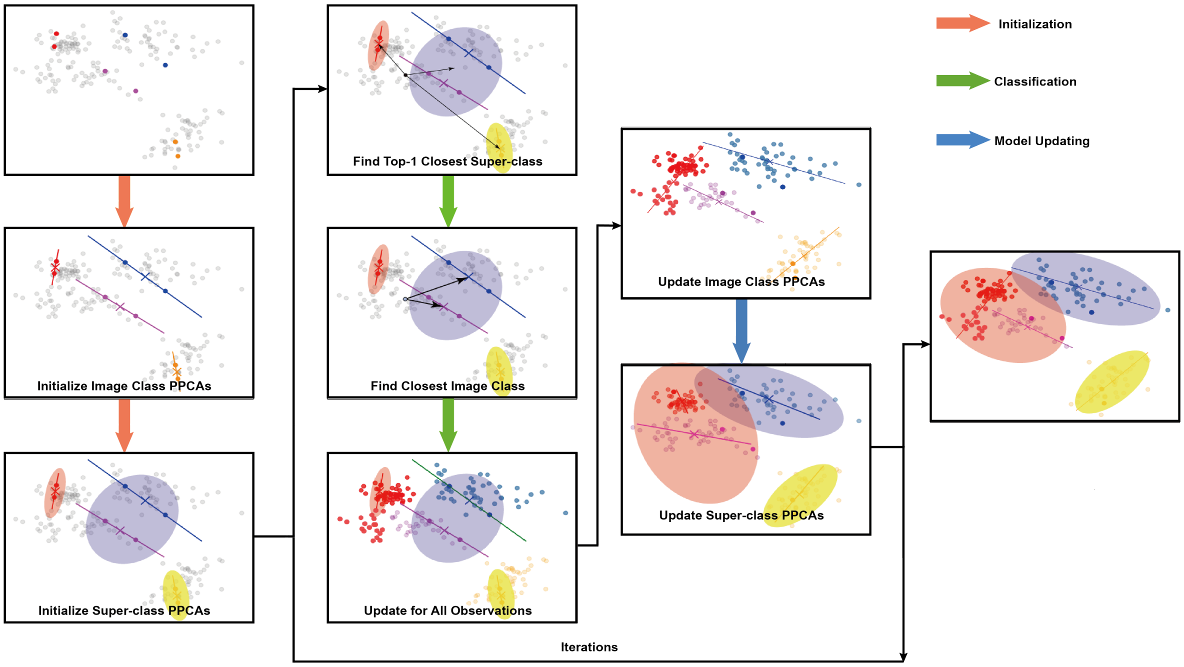

Figure 4.2: Flowchart of H$k$-PPCAs: Four image classes and three super-classes, number of candidate super-classes T=1. (The colored points are labeled or classified observations. The gray points are unlabeled.)

2017), considered a classical metric-based method, is thought to generate an embedding where data points cluster around a single prototype representation for each class. DeepEMD (Zhang et al., 2020) proposes adopting Earth Mover's Distance as a metric to calculate the structural distance between dense image representations, determining the similarity between images. For optimization-based methods, MAML (Finn et al., 2017) learns an optimization method to rapidly learn a classifier for novel classes along the fast gradient direction. Ravi and Larochelle (2016) formulates the parameter update into an LSTM and accomplishes this through a meta-learner.

Regarding transfer-learning-based methods, a model is typically pre-trained on a large amount of data from base classes and then utilizes the pre-trained model to recognize novel classes in few-shot classification. More specifically, Qi et al. (2018) proposes a new method, weighted imprinting, which directly sets the final layer weights from novel training examples during few-shot learning. Chen et al. (2019) investigates and demonstrates that transfer-learning-based methods can achieve competitive performance similar to meta-learning methods.

With the development of large language models in recent years (Vaswani et al., 2017; Dosovitskiy et al., 2020), the transfer-learning-based model with Transformer backbone has achieved significant achievements in large-scale dataset (Riquelme et al., 2021; Singh et al., 2023). Riquelme et al. (2021) proposes Vision Mixture of Experts, a distributed sparsely-activated Transformer model (Dosovitskiy et al., 2020) for vision, and showed remarkable scalability on various datasets. Singh et al. (2023) introduces a pre-pretraining stage based on MAE (He et al., 2022), which significantly improves the performance on various datasets.

**Semi-Supervised Few-shot Learning.** Semi-Supervised Few-Shot Learning (SSFSL) is an overlapped set of Semi-Supervised Learning (SSL) and Few-Shot Learning (FSL). There has been a growing interest in leveraging unlabeled data to enhance the accuracy of few-shot learning. However, directly applying SSL methods to the few-shot setting often yields subpar results due to the extremely small number of labeled observations. To address the challenges of SSFSL, Ren et al. (2018) adapted Prototypical Networks (Snell et al., 2017) and incorporated unlabeled samples in prototype generation. Liu et al. (2018) proposed TPN to learn the manifold structure of labeled data for label propagation from labeled data to unlabeled data. Recent SSFSL methods, including Wang et al. (2020); Lazarou et al. (2021); Huang et al. (2021), have been introduced to skillfully predict pseudo-labels for unlabeled data and augment the few-shot set subsequently in classification.

**Hierarchical Clustering.** Hierarchical clustering is an unsupervised clustering technique designed to construct a hierarchy of clusters (Day and Edelsbrunner, 1984). Two primary categories within hierarchical clustering are agglomerative (Gowda and Krishna, 1978) and divisive methods (Samek et al., 2017), representing the construction of hierarchies through bottom-up and top-down approaches, respectively.

Several endeavors have explored hierarchical clustering in the field of computer vision. For instance, Zweig and Weinshall (2007) manually crafted a hierarchical tree that amalgamates image classifiers from different levels into a single higher-level classifier for classification. Marszalek and Schmid (2007) integrated prior information about inter-class relationships derived from the word semantics of image labels, constructing a hierarchy of visual discriminative classifiers inherited from the word hierarchy. Jia et al. (2013) introduced a hierarchical framework learned from the confusion matrix of classes using a Bayesian generalization model, capable of determining an appropriate hierarchical level of generalization from a set of images. Srivastava and Salakhutdinov (2013) enhanced neural networks with a hierarchical class prior on the last layer of weights, proving beneficial when the training set is small, as classes with few examples can access shared features belonging to the super-class.

## 4.3 Hierarchical $k$-PPCAs

This section presents the proposed hierarchical method named Hierarchical $k$-PPCAs (H$k$-PPCAs), which uses probabilistic PCA to incorporate the semantic information behind image classes. First, for the completeness of the introduction, an overview of the probabilistic PCA model is reviewed and the parameter estimation method, k-means clustering using the Mahalanobis distance, is explained. Then the proposed algorithm for semi-supervised large-scale learning is introduced in detail, including running-average-based parameter adaptation and a discussion on computation complexity.

**Probabilistic PCA Representation.** As introduced in Sec. 2.3.1, Probabilistic PCA (PPCA) (Tipping and Bishop, 1999b) models a data cluster with a Gaussian distribution proxy in a lower-dimensional subspace. The $d$-dimensional observable variable $\mathbf{x} \in \mathbb{R}^d$ is modeled by a lower $q$-dimensional latent variable $\mathbf{t} \in \mathbb{R}^q$ as expressed in Eq. (2.1) and the conditional distribution of $\mathbf{x}$ given the latent variable $\mathbf{t}$ is Eq. (2.2).

Similar to the derivation in $k$-PPCAs, the singular value decomposition (SVD) is applied to the sample covariance matrix and the covariance matrix is reconstructed by the principal components from SVD, as described in Eq. (2.4). We have

$$\hat{\mathbf{\Sigma}}^{(PPCA)} = \mathbf{L}^T \mathbf{D} \mathbf{L} + \lambda \mathbf{I}_d. \tag{4.1}$$

**Hierarchical $k$-Means Classification with Probabilistic PCAs (H$k$-PPCAs).** In the proposed approach, each image class is modeled by PPCA, and similar image classes are viewed as sub-classes from a common super-class, also modeled by PPCA, so that a two-level taxonomy hierarchical structure is formed. Our approach performs the classification clustering by $k$-MeansLloyd (1982) with the Mahalanobis distance, and generates the super-class clusters by $k$-Means with Kullback-Leibler divergence.

**Feature Extraction.** The feature extractor $\boldsymbol{f} : \Omega \to \mathbb{R}^d$ projects input images to informative numerical features. The proposed work utilized a pretrained and frozen self-learning encoder as the feature extractor. More details will be described in Sec. 5.1.2.

**Algorithm 3** H$k$-PPCAs Initialization

```
Input: Labeled dataset $D^{(l)}_{train} = \{D^{(l)}_1, ..., D^{(l)}_k, ..., D^{(l)}_K\}$ for all image-classes $k = 1, \cdots, K$. The number
    of super-classes $M$.
Output: Initial image-class PPCAs $\theta^{(c)}_k = (\boldsymbol{\mu}^{(c)}_k, \mathbf{L}^{(c)}_k, \mathbf{S}^{(c)}_k)$ for each image-class $k = 1, \cdots, K$ and super-
    class PPCAs $\theta^{(s)}_m = (\boldsymbol{\mu}^{(s)}_m, \mathbf{L}^{(s)}_m, \mathbf{S}^{(s)}_m)$ for each super-class $m = 1, \cdots, M$.
for each class $k = 1, \cdots, K$ do
    Obtain $\boldsymbol{\mu}^{(c)}_k$ and $\mathbf{\Sigma}^{(c)}_k$ from $D^{(l)}_k$ using Eq. (2.12) and Eq. (4.3) respectively.
    Obtain $\mathbf{S}^{(c)}_k$ from $\mathbf{V}_k \mathbf{S}_k \mathbf{V}^T_k = \mathbf{\Sigma}^{(c)}_k$ using SVD
    Obtain $\mathbf{L}^{(c)}_k$ as the first $q$ columns of $\mathbf{V}_k$
end for
Initialize super-class PPCAs $\theta^{(s)}_m = (\boldsymbol{\mu}^{(s)}_m, \mathbf{L}^{(s)}_m, \mathbf{S}^{(s)}_m)$ with Algorithm 4
Update super-class PPCAs with Algorithm 5
```

**H$k$-PPCAs Initialization.** Since the proposed work is designed to solve the semi-supervised task, the training dataset $D_{train}$ includes the labeled subset $D^{(l)}_{train}$ and the unlabeled subset $D^{(u)}_{train}$.

The initialization of the H$k$-PPCAs system includes two parts: image-class PPCAs and super-class PPCAs. For image classes, the sample mean and sample covariance of labeled observations are calculated for each image class,

$$\hat{\boldsymbol{\mu}}^{(c)}_k = \frac{\sum_{\{i|y_i=k\}} \boldsymbol{x}_i}{n_k}, \tag{4.2}$$

$$\hat{\boldsymbol{\Sigma}}^{(c)}_k = \frac{\sum_{\{i|y_i=k\}} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)^T}{n_k - 1}, \tag{4.3}$$

where $n_k$ is the number of labeled observations in the $k$-th image class PPCA cluster. The superscript $(c)$ indicates the parameters for image classes and the superscript $(s)$ in the following section denotes the parameters for super classes. SVD is applied to the sample covariance afterwards and the PPCA parameters $\theta^{(c)}_k = (\boldsymbol{\mu}^{(c)}_k, \mathbf{L}^{(c)}_k, \mathbf{S}^{(c)}_k)$ are obtained for each image class. The algorithm of overall H$k$-PPCAs initialization is described in Algorithm 3.

In respect of super-class PPCAs, the $k$-Means++Arthur and Vassilvitskii (2007) initialization is adopted for a better clustering performance and a faster convergence. Specifically, the Bhattacharyya distance is employed as the distance measure of two distributions, e.g. for $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ and $\mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$,

$$D_{ij} = \frac{1}{8}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) + \frac{1}{2} \ln \frac{|\boldsymbol{\Sigma}|}{\sqrt{|\boldsymbol{\Sigma}_i||\boldsymbol{\Sigma}_j|}}, \tag{4.4}$$

where $\boldsymbol{\Sigma} = \frac{\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j}{2}$. The algorithm of super-class initialization is detailed in Algorithm 4.

After the initialization of super-classes, the super-classes need to be updated to include more image classes and form the two-level hierarchical structure. The proposed method utilizes the $k$-Means with Kullback-Leibler (KL) divergence as the distance measure between image classes and super-classes, both of which are modeled as Gaussian distributions, where the covariance is reconstructed by PPCA using Eq. (4.1). The KL divergence reflects how a probability distribution $P$ is different from a reference distribution. In our proposed method, we assume the super-class is the reference distribution $Q = \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$, a hypothetical cluster of image classes, and the image classes are the observed distribution $P = \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$.

$$\begin{aligned} D_{KL}(p||q) = \frac{1}{2}[\log \frac{|\boldsymbol{\Sigma}_q|}{|\boldsymbol{\Sigma}_p|} &+ (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^T \boldsymbol{\Sigma}_q^{-1} (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q) \\ &+ \mathrm{Tr}\{\boldsymbol{\Sigma}_q^{-1} \boldsymbol{\Sigma}_p\} - d]. \end{aligned} \tag{4.5}$$

For each image class, the KL divergence from each of the super-classes will be measured and compared, and the super-class with the least KL divergence will be the assigned super-class label of this image class.

**Algorithm 4** Super-class PPCAs initialization with $k$-Means++

```
Input: Image-class PPCAs $\theta^{(c)} = \{\theta_1^{(c)}, \cdots, \theta_k^{(c)}, \cdots, \theta_K^{(c)}\}$ for all image-classes $k = 1, \cdots, K$, and number $M$ of super-classes.
Output: Super-class PPCAs $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ for each super-class $m = 1, \cdots, M$.
for each class $k = 1, \cdots, K$ do
    reconstruct $\hat{\boldsymbol{\Sigma}}_k^{(c,\ PPCA)}$ using Eq. (4.1)
end for
Randomly pick an image-class $(\boldsymbol{\mu}_k^{(c)}, \hat{\boldsymbol{\Sigma}}_k^{(c,\ PPCA)})$ as the first centroid $C_1$
for $m = 2, \cdots, M$ do
    Calculate Bhattacharyya distance between the last selected centroid $C_{m-1}$ using Eq. (4.4)
    Construct the probability distribution of image classes where the mass function is proportional to the distance from each image class to its closest centroid from $\{C_1, \cdots, C_{m-1}\}$
    Sample the image class $(\boldsymbol{\mu}_j^{(c)}, \hat{\boldsymbol{\Sigma}}_j^{(c,\ PPCA)})$ as the new centroid $C_m = (\boldsymbol{\mu}_m^{(s)}, \boldsymbol{\Sigma}_m^{(s)})$ from the generated distribution
end for
for $m = 1, \cdots, M$ do
    Obtain $\mathbf{S}_m^{(s)}$ from $\mathbf{V}_m \mathbf{S}_m \mathbf{V}_m^T = \boldsymbol{\Sigma}_m^{(s)}$ using SVD
    Obtain $\mathbf{L}_m^{(s)}$ as the first $q$ columns of $\mathbf{V}_m$
end for
```

After all image classes receive their assigned label, the super-class mean and covariance can be obtained from its assigned image class. Similar to the process of accessing PPCA parameters in the image class, SVD is applied to the covariance and super-class PPCA parameters $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ are obtained. The process is explained in Algorithm 5.

**Classification and Parameter Estimation.** When using PPCA for classification, the image classes are assumed to be mostly separable from each other in the feature space. Again, similar to $k$-PPCAs, image classes are modeled as different Gaussian distributions where the covariance is remodeled by PPCA, for example, the data belonging to the $k$-th image class follow the Gaussian distribution with mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$. The negative log-likelihood of observation $\mathbf{x}$ for a given class $k$ is shown in Eq. (2.6) and further simplified as Mahalanobis distance in Eq. (2.8)

The Mahalanobis distance is a measure of the distance between a point and a distribution by the unit of standard deviation of this distribution. The benefit of employing Mahalanobis distance in the proposed method is it considers the correlations among features and the shape information of the image classes.

Given PPCA parameters $\theta = (\boldsymbol{\mu}, \mathbf{L}, \mathbf{S})$ and denoting by $\mathbf{s} \in \mathbb{R}^q$ the vector containing the first $q$ diagonal elements of $\mathbf{S}$, the score (2.8) can be more efficiently computed using Theorem 1.

**Algorithm 5** Super-class PPCAs update with $k$-Means

```
Input: Image-class PPCAs $\theta^{(c)} = \{\theta_1^{(c)}, \cdots, \theta_k^{(c)}, \cdots, \theta_K^{(c)}\}$ for all image-classes $k = 1, \cdots, K$, super-class PPCAs $\theta^{(s)} = \{\theta_1^{(s)}, \cdots, \theta_m^{(s)}, \cdots, \theta_M^{(s)}\}$ for each super-class $m = 1, \cdots, M$, number $M$ of super-classes, and number $n_{iter}^{(s)}$ of iterations in super-class updates
Output: Super-class PPCAs $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ for each super-class $m = 1, \cdots, M$.
for i=1 to $n_{iter}^{(s)}$ do
    Initialize super-class RAVEs $R_m^{(s)}$ for $m = 1, \cdots, M$
    for each image class $k = 1, \cdots, K$ do
        Use Eq. (4.5) to compute
            $$d_m{=}D_{KL}(\mathcal{N}(\boldsymbol{\mu}_k^{(c)}, \boldsymbol{\Sigma}_k^{(c,PPCA)})||\mathcal{N}(\boldsymbol{\mu}_m^{(s)}, \boldsymbol{\Sigma}_m^{(s,PPCA)}))$$
        for all $m \in \{1, ..., m\}$.
        Obtain super-class label $m^* = \operatorname{argmin}_m d_m$.
        Add image class $(\boldsymbol{\mu}_k^{(c)}, \hat{\boldsymbol{\Sigma}}_k^{(c,\ PPCA)})$ to $R_{m^*}^{(s)}$
    end for
    for $m = 1, \cdots, M$ do
        Obtain $\boldsymbol{\mu}_m^{(s)}$ and $\boldsymbol{\Sigma}_m^{(s)}$ from $R_m$ using Eq. (2.17)
        Obtain $\mathbf{S}_m^{(s)}$ from $\mathbf{V}_m \mathbf{S}_m \mathbf{V}_m^T = \boldsymbol{\Sigma}_m^{(s)}$ using SVD
        Obtain $\mathbf{L}_m^{(s)}$ as the first $q$ columns of $\mathbf{V}_m$
        Reconstruct $\hat{\boldsymbol{\Sigma}}_m^{(s,\ PPCA)}$ using Eq. (4.1)
    end for
end for
```

Our approach performs classification by k-Means with the Mahalanobis distance. Considering the number of classes could be tens of thousands or more in a large-scale dataset, distance calculation becomes computationally expensive. In the proposed method, we leverage the hierarchical structure and search the image classes only in the closest $T$ candidate super-classes for each observation, which shrinks the search range to a manageable scale. Specifically, the Mahalanobis distances to all $M$ super-classes are calculated for each unlabeled observation $\mathbf{x}_i \in \mathbb{R}^d$, and the closest $T$ super-classes are selected as candidates $S^*$. Based on the hierarchy mapping between super-classes and image classes, a set of image classes within $S^*$ can be formed, denoted as the candidate image classes $C^*$. The scale of the candidate image class $|C^*|$ is much smaller than the total number of image classes $K$ if the image classes are evenly distributed in the super-classes so that the distance calculation can be significantly improved by this hierarchical design.

Then the unlabeled observation $\mathbf{x}_i$ will be assigned to the closest candidate image class,

$$y_i = \underset{k \in C^*}{\operatorname{argmin}}\, r^{(c)}(\mathbf{x}) = \underset{k \in C^*}{\operatorname{argmin}} (\mathbf{x}_i - \boldsymbol{\mu}_k^{(c)})^T \left[\boldsymbol{\Sigma}_k^{(c)}\right]^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k^{(c)}). \tag{4.6}$$

In addition, all labeled observations will be directly assigned to the image class corresponding to their labels. For better robustness from outliers, the observations with obviously large scores will be identified as extreme values and removed from the parameter update step. Under the assumption that each of the $d$ variables in a feature vector follows a normal distribution, the Mahalanobis distance will follow a $\chi_d^2$ distribution. Therefore in our proposed method, any observations whose score is greater than $\chi^2_{d,0.975}$ are labeled as extreme values, and this criterion approximately filters out 2.5% observations with the greatest Mahalanobis distance in the image class.

After the assignment of all observations, the sample mean and sample covariance for each image class cluster ($\hat{\boldsymbol{\mu}}_k^{(c)}$,$\hat{\boldsymbol{\Sigma}}_k^{(c)}$) are computed, and SVD is applied to the covariance to obtain the updated image class PPCA parameters $\theta_k^{(c)} = (\boldsymbol{\mu}_k^{(c)}, \mathbf{L}_k^{(c)}, \mathbf{S}_k^{(c)})$. Similar to the initialization process, the super-class PPCA parameters are obtained from the image class PPCAs using Algorithm 4 and Algorithm 5. The complete algorithm is detailed in Algorithm 6.

**Online Update of Covariance with Running Averages.** In practice, lines 3-18 from Algorithm 6, which assign labels to the observations based on the Mahalanobis distance, are performed by mini-batches due to the large scale of the dataset and GPU memory limitations. However, the mean and the covariance are calculated (Eq. (4.2) and (4.3)) after assigning all the observations and the mini-batches have been discarded. In our proposed method, the running averages framework (RAVE) (Sun et al., 2024) in Sec. 2.3.4

**Algorithm 6** H$k$-PPCAs

```
Input: Image-class PPCAs $\theta_k^{(c)} = (\boldsymbol{\mu}_k^{(c)}, \mathbf{L}_k^{(c)}, \mathbf{S}_k^{(c)})$ for $k = 1, \cdots, K$, super-class PPCAs $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ for $m = 1, \cdots, M$, training dataset $D_{train} = \{D_{train}^{(l)}, D_{train}^{(u)}\}$ including labeled set $D_{train}^{(l)}$ and unlabeled set $D_{train}^{(u)}$, number $T$ of candidate super-classes, number $n_{iter}$ of iterations.
Output: Updated image-class PPCAs $\theta_k^{(c)} = (\boldsymbol{\mu}_k^{(c)}, \mathbf{L}_k^{(c)}, \mathbf{S}_k^{(c)})$ $k = 1, \cdots, K$ and updated super-class PPCAs $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ for $m = 1, \cdots, M$.
for i=1 to $n_{iter}$ do
    Initialize image class RAVEs $R_k^{(c)}$ for $k = 1, \cdots, K$
    for each observation $\mathbf{x} \in D_{train}$ do
        if $\mathbf{x} \in D_{train}^{(l)}$ then
            Set $k^* = y$ as the observation label $y$
        else
            Compute scores $r_m^{(s)} = r(\mathbf{x}, \boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)}), m = 1, \cdots, M$
            Obtain top-$T$ candidate super-classes $S^* = \{S^{(1)}, \cdots, S^{(T)}\}$ with smallest scores $(r^{(s,1)}, \cdots, r^{(s,T)})$
            Obtain candidate image classes $C^* = \{k | C_k \in S^*\}$
            Compute scores $r_k^{(c)} = r(\mathbf{x}, \boldsymbol{\mu}_k^{(c)}, \mathbf{L}_k^{(c)}, \mathbf{S}_k^{(c)})$ for $k \in C^*$
            Obtain label $k^* = \mathrm{argmin}_{k \in C^*} r_k^{(c)}$
        end if
        if $r_k < \chi^2_{d,\tau}$ then
            add $\mathbf{x}$ to $R_{k^*}^{(c)}$ using Eq. (2.18)
        end if
    end for
    for $k = 1, \cdots, K$ do
        Obtain $\boldsymbol{\mu}_k^{(c)}$ and $\boldsymbol{\Sigma}_k^{(c)}$ from $R_k^{(c)}$ using Eq. (2.17)
        Obtain $\mathbf{S}_k^{(c)}$ from $\mathbf{V}_k \mathbf{S}_k \mathbf{V}_k^T = \boldsymbol{\Sigma}_k^{(c)}$ using SVD
        Obtain $\mathbf{L}_k^{(c)}$ as the first $q$ columns of $\mathbf{V}_k$
    end for
    Initialize super-class PPCAs $\theta_m^{(s)} = (\boldsymbol{\mu}_m^{(s)}, \mathbf{L}_m^{(s)}, \mathbf{S}_m^{(s)})$ with Algorithm 4
    Update super-class PPCAs with Algorithm 5
end for
```

is employed again in H$k$-PPCAs to compute and update the mean and the covariance incrementally during screening mini-batches.

For the new batch $B = \{1, ..., n_B\}$, $\mathbf{S}_x^{(B)} = \mathbf{S}_x^{(n_B)}$ and $\mathbf{S}_{xx}^{(B)} = \mathbf{S}_{xx}^{(n_B)}$, the target running averages in Eq. (2.16) and Eq. (2.18) can be updated as follows:

$$\mathbf{S}_x^{(n+n_B)} = \frac{n}{n+n_B}\mathbf{S}_x^{(n)} + \frac{n_B}{n+n_B}\mathbf{S}_x^{(n_B)} \tag{4.7}$$

and similarly for $\mathbf{S}_{xx}^{(n+n_B)}$.

**Complexity Analysis of H$k$-PPCAs.** The computation cost can be estimated by the number of PPCA neurons involved in the classification and parameter estimation. Let $K$, $M$, and $T$ be the number of image classes, number of super-classes, and number of candidate super-classes respectively, then there are $KT/M$ candidate image classes on average within one super-class, assuming the image classes are uniformly distributed in super-classes. Hence $M + KT/M$ scores need to be calculated for each observation.

When compared with a non-hierarchical design (flat design), in which $K$ scores need to be generated for each observation, a metric speed-up can be designed to measure the improvement from a flat design to the hierarchical design,

$$\text{speed-up} = \frac{K}{M + KT/M} = \frac{1}{M/K + T/M} \tag{4.8}$$

From the Eq. (4.8) above, the maximum speed-up can be obtained when $M = \sqrt{KT}$, where the speed improvement can reach $O(\sqrt{K/T})$.

Furthermore, let $N$, $q$, and $d$ be the sample size, number of principal components, and dimension of the features. Based on the analysis of optimum $M$ above, the complexity of calculating Mahalanobis distance by Eq. (2.9) from one observation to one class is $O(q^2d)$, then for $N$ observations, $\sqrt{KT}$ candidate super-classes and $\sqrt{KT}$ candidate image classes, the complexity of assigning observations is $O(q^2dN\sqrt{KT})$. For covariance estimation, the time complexity is $O(d^2N)$. The complexity of SVD to each covariance is $O(d^3)$, then for $\sqrt{KT}$ image classes, the complexity is $O(d^3\sqrt{KT})$. So the overall complexity of $k$-PPCAs is $O(q^2dN\sqrt{KT} + d^2N + d^3\sqrt{KT})$.

# CHAPTER 5

# EXPERIMENTAL EVALUATION AND DISCUSSION FOR HIERARCHICAL K-PPCAS

## 5.1 Experiments

The proposed two-level hierarchical design and a non-hierarchical flat design are evaluated and compared on two popular large-scale datasets, ILSVRC-2012 (ImageNet-1k) and ImageNet-10k datasets. The flat design is a one-level mixture of PPCAs where the classification is performed on all image classes $k \in \{1, \cdots, K\}$ without using any super-classes.

### 5.1.1 Datasets

**ImageNet-1k (Russakovsky et al., 2015).** ImageNet-1k is one of the most popular large-scale datasets for classification which contains 1,281,167 training and 50,000 validation images from 1000 object classes. There are about 1,300 training images and 50 validation images in each class.

**ImageNet-10k.** ImageNet-10k is the subset of the whole ImageNet (Russakovsky et al., 2015) that includes 10,450 classes and more than nine million images. For each category, there are at least 450 training images and 50 testing images.

### 5.1.2 Implementation Details

**Feature Extraction.** As mentioned in Sec. 4.3, a pretrained and frozen image encoder of a flexible and generic contrastive learning method, CLIP (Radford et al., 2021), is adopted as the feature extractor for the proposed method. CLIP maximizes the similarity between the image features and the associated text embeddings. CLIP pretrained the image encoders with different backbones, e.g. ResNet-50x4, ResNet-50, etc, on a wide variety of images from the internet. In our experiments, the ResNet-50x4 encoder is adopted as the backbone of the feature extractor.

For image preprocessing on the ImageNet datasets, we follow the settings of the pretrained CLIP encoder. The short edge of the image is resized to 288 while maintaining the original aspect ratio, followed by central cropping to obtain a square image as input for the feature extractor. The transformed size is 288 for CLIP backbones.

**Discussion about CLIP and ImageNet.** We would like to discuss the training set of CLIP because our results on ImageNet-1k and ImageNet-10k can be less reliable if there is an overlapping between the training set of CLIP and ImageNet. However, even though the detailed data source is not provided, they mentioned the dataset is created from various publicly available sources on the Internet, and more convincingly, the CLIP can be used for zero-shot transfer learning and is evaluated on ImageNet-1k in their paper. In addition, there is an existing method (Nakata et al., 2022) that applied CLIP to evaluate a class-incremental learning task on ImageNet-1k.

**Raw-Observation Based Super-Class.** In Algorithm 6, super-classes are estimated by clustering Gaussian distributions of image classes, in which the covariance is remodeled by PPCA. Another method of forming the hierarchical structure is also evaluated by clustering the raw observations that represent each image class. A subset of the observations are sampled from the image classes, and then these raw observations, as prototypes of their image class, will be assigned to the closest super-class. Because the raw observations from the same image class can be assigned to different super-classes, each image class could be shared by multiple super-classes, which is meaningful since a single image class can be categorized from different aspects. The super-classes are updated afterward as described in Algorithm 5. Since the distance measures a raw observation to a super-class Gaussian distribution, the KL divergence in Algorithm 5 is replaced by the Mahalanobis distance using Eq. (2.8).

In our experiment, 20 raw observations are sampled from each image class. Two different sampling methods are evaluated, one is to randomly sample from each image class, and the other one is to pick the observations with the smallest scores to their image class centers, which makes the selection having higher confidence represent their image class. The comparison is provided in Sec. 5.2.1.

**Other Details.** The value of the PPCA parameter $\lambda$ in Eq. (4.1) and Eq. (2.9) is set to $\lambda = 0.01$. The number of labeled data per image class is $l = 2$, the number of candidate super-classes is $T = 4$, and the number of principal components is $q = 2$. For both benchmarks, the model is trained for 10 epochs. The speed-up in Eq. (4.8) is employed as the efficiency metric in the experiments. All experiments were performed on an RTX 3060 GPU.

### 5.1.3 Results

We evaluated the flat (non-hierarchical) $k$-Means, which applied Euclidean distance in Eq. (2.6), flat $k$-PPCAs, the proposed H$k$-PPCAs with PPCA-based clustering, and the proposed method with raw-observation clustering on ImageNet-1k and ImageNet-10k. To the best of our knowledge, we are not aware of other

Table 5.1: H$k$-PPCAs evaluation on ImageNet-1k and ImageNet-10k with number of super-classes $M = 50$ and $M = 200$ respectively. The flat design in Sec. 4.3 is compared with H$k$-PPCAs with PPCA-based super-class clustering and raw-observation clustering in terms of accuracy and efficiency. The efficiency metric is the speed-up defined in Sec. 4.3.

| Method | ImageNet-1k | | ImageNet-10k | |
|---|---|---|---|---|
| | Accuracy(%) | Speed-up | Accuracy(%) | Speed-up |
| Flat $k$-Means | 51.17 | 1.0 | 14.98 | 1.0 |
| Flat $k$-PPCAs | 52.47 | 1.0 | 16.61 | 1.0 |
| H$k$-PPCAs | 51.19 | 4.8 | 15.62 | 19.9 |
| Random raw observations | 52.64 | 2.5 | 16.72 | 8.5 |

Table 5.2: Super-class construction with raw-observation clustering: randomly sampled raw observations vs high confidence raw observations.

| Method | ImageNet-1k | | ImageNet-10k | |
|---|---|---|---|---|
| | Accuracy(%) | Speed-up | Accuracy(%) | Speed-up |
| Random sampling | 52.13 | 1.7 | 16.5 | 6.1 |
| High confidence raw obs. | 52.09 | 2.1 | 16.43 | 8.6 |

methods evaluated based on the same backbone and benchmarks, so only our results are reported in Tab. 5.1. The non-hierarchical $k$-Means obtains an accuracy of 51.17% on ImageNet-1k, and the flat-designed $k$-PPCAs improves the accuracy to 52.47%. The proposed hierarchical method, H$k$-PPCAs with PPCA-clustered super-classes, achieves an accuracy of 51.49% which is a little inferior to the flat $k$-PPCAs, but instead, receives a 4.8 speed-up compared to the flat design. As for the proposed method using raw-observation-clustered super-classes and augmentation, it has an accuracy, of 52.64%, which slightly improves the flat design, and obtains a speed-up of 2.5. Regarding ImageNet-10k, we can observe similar results. The non-hierarchical $k$-Means and $k$-PPCAs achieve the accuracy of 14.98% and 16.61% respectively on 10,450 categories. The proposed H$k$-PPCAs with PPCA-clustered super-classes shows a slightly inferior accuracy of 15.62% but has a speed-up of 19.9 compared to the flat design. After clustering the super-classes by randomly sampled raw data, the accuracy is further improved to 16.72% having a speed-up of 8.5.

## 5.2 Discussion

### 5.2.1 Ablation Study

The ablation study will evaluate the importance of the selection method for the observations (raw observations) that will be used as prototypes for the image classes and of data augmentation for increasing the pool of labeled observations.

**Raw-Observation Selection.** As explained in Sec. 5.1.2, the raw observations are selected as the prototypes from each of the image classes, and the super-classes are obtained by clustering these selected raw observations. Two different methods of observation selection are evaluated. The first one is the raw observations are randomly sampled from all labeled and unlabeled data in each image class. The other method is to choose the observations with the highest confidence (smallest scores) within each image class, which makes the selected raw data close to the center of their image class.

The selected observations in the second method are more clustered around the image class center, so it is less possible to observe observations from the same image class captured by different super-classes, therefore, the candidate image classes are fewer than the first method and the speed-up is expected to be higher than the first method.

The results based on the two selection methods are provided in Tab. 5.2. From the comparison, we can see the accuracy of the random sampling method slightly surpasses the second method on both benchmarks. As we expected above, a higher speed-up is obtained by using the more clustered low-score raw observations in the second method.

**Data Augmentation on Labeled Raw Observations.** Since the few-shot setting limits the definite image class information from the labeled observations, data augmentation is applied to generate modified copies of the labeled data and we put more weight on the labeled raw observations while estimating the super-classes.

Specifically, in ImageNet-1k, ten augmented observations are generated from each of the labeled data, hence there are 20 augmented labeled observations and two originally labeled observations per image class. Then, during the estimation of super-classes, the collection of raw observations is formed by 12 labeled and eight unlabeled observations in two ways, random sampling and high confidence selection, as mentioned in Sec. 5.1.2. For ImageNet-10k, due to its large scale, two augmented images per observation are generated hence there are 4 augmented labeled observations and two originally labeled observations per image class. The raw observation collection is formed by 6 labeled and 14 unlabeled observations. The evaluation results are shown in Tab. 5.3.

An accuracy increase of 0.54% and 0.55% for ImageNet-1k and 0.3% and 0.29% for ImageNet-10k are observed respectively for random sampling and high confidence selection in Tab. 5.3. If compared with the results in Tab. 5.1, we can find the methods with augmented raw observations surpass the flat design in accuracy and obtain a speed increase at the same time.

Table 5.3: Data augmentation to raw observations. The collection of raw observations consists of 12 labeled observations and 8 unlabeled observations.

| Method | Augmentation | ImageNet-1k | | ImageNet-10k | |
|---|---|---|---|---|---|
| | | Accuracy(%) | Speed-up | Accuracy(%) | Speed-up |
| Random Sampling | N | 52.13 | 1.7 | 16.5 | 6.1 |
| | Y | 52.67 | 2.0 | 16.8 | 6.1 |
| High confidence raw obs. | N | 52.09 | 2.1 | 16.43 | 8.6 |
| | Y | 52.64 | 2.5 | 16.72 | 8.5 |

### 5.2.2 Conclusion

Chapters Four and Five introduced a novel framework aimed at addressing the challenges of few-shot large-scale semi-supervised classification. The proposed method, Hierarchical $k$-Probabilistic Principal Component Analyzers (H$k$-PPCAs), is designed to efficiently handle large-scale datasets by leveraging a hierarchical classification structure based on a set of PPCA-modeled Gaussian distributions. This framework builds upon the strengths of probabilistic principal component analysis (PPCA) by modeling class covariance structures with a small number of principal components, thereby capturing essential variations in the data while significantly reducing computational overhead.

The key innovation of H$k$-PPCAs lies in its hierarchical design, which enables the model to classify an observation by first narrowing down the candidate classes through a super-class hierarchy. Instead of searching through all possible image classes, the method focuses only on the most likely super-classes for a given observation, reducing the search space to the classes within those super-classes. This approach significantly improves efficiency, reducing the computational complexity from $O(K)$ for K classes in a flat classification model to $O(\sqrt{K})$ in the hierarchical model. Such a reduction in complexity makes the H$k$-PPCAs framework highly scalable and practical for large-scale datasets, where the number of classes can be prohibitively large.

The framework's hierarchical nature not only reduces computation time but also ensures that classification remains accurate by organizing classes based on semantic similarity. The use of PPCA-based Gaussian distributions allows the model to effectively discriminate between different classes, even when the data is sparse or noisy, as is often the case in few-shot learning scenarios.

Extensive experiments were conducted to validate the effectiveness of H$k$-PPCAs on two widely-used large-scale benchmarks: ImageNet-1k and ImageNet-10k. These experiments demonstrated that the hierarchical structure of H$k$-PPCAs provides both computational efficiency and robust classification performance, making it an ideal solution for large-scale semi-supervised classification tasks. The results confirmed that the proposed method not only scales well with an increasing number of classes but also maintains high classification accuracy, even with limited labeled samples, which is a key challenge in few-shot learning.

# CHAPTER 6

# SUMMARY AND FUTURE RESEARCH

## 6.1 Summary

This dissertation focused on two challenging yet highly practical problems: Semi-Supervised Few-Shot Class Incremental Learning (SSFSCIL) and Large-Scale Semi-Supervised Few-Shot Learning (SSFSL).

Chapter Two introduced an efficient probabilistic classifier, $k$-PPCAs, which leverages a pretrained self-supervised feature extractor to address the issue of catastrophic forgetting in Few-Shot Class Incremental Learning tasks. The $k$-PPCAs framework models classes using a mixture of probabilistic principal component analyzers, enabling it to effectively capture and differentiate class variance in the feature space under the FSCIL setting. This metric-based approach also allows for the identification of out-of-distribution samples by assigning them a low-confidence score, classifying them as ”unknown.”

Extensive experiments reported in Chapter Three demonstrate the effectiveness of $k$-PPCAs on several well-known benchmarks, including CUB200, CIFAR100, and miniImageNet. Additionally, the method is evaluated on the large-scale ImageNet-1k dataset, which had not previously been tested in SSFSCIL due to its size, further validating the scalability and robustness of the approach.

The second framework presented in this dissertation, discussed in Chapter Four, addresses the problem of Large-Scale Semi-Supervised Few-Shot Classification. The proposed method, H$k$-PPCAs, is a hierarchical classifier based on a collection of PPCA-modeled Gaussian distributions. By modeling the essential variations of data with fewer principal components, this approach reduces both computational cost and model complexity. H$k$-PPCAs classifies observations by first narrowing down the candidate super-classes before focusing on the specific image classes within those super-classes, reducing the overall search space. This hierarchical structure decreases the computational complexity from $O(K)$ to $O(\sqrt{K})$ for K classes, making it feasible for use with large-scale datasets.

Chapter Five presents experimental results that validate the effectiveness of H$k$-PPCAs on two widely-used large-scale benchmarks, ImageNet-1k and ImageNet-10k, confirming the practicality of the proposed method for large-scale semi-supervised classification tasks.

## 6.2 Future Research

The proposed $k$-PPCAs and H$k$-PPCAs classifiers are built on top of a pretrained feature extractor, meaning that the performance of these classifiers is heavily dependent on the quality and specific purpose of the feature extractor used. For example, if the feature extractor is trained to recognize broad, coarse-grained categories, such as "bird," it may pose challenges when applied to tasks requiring finer distinctions, such as classifying different bird subspecies. This limitation suggests the need to explore an end-to-end PPCA-based classification algorithm in future work, where the feature extraction and classification processes are optimized together.

Additionally, we plan to extend our approach to even larger datasets, such as the full ImageNet dataset, which contains over 20,000 classes, to further evaluate the scalability and robustness of the proposed methods.

# BIBLIOGRAPHY

Ahmad, T., Dhamija, A. R., Cruz, S., Rabinowitz, R., Li, C., Jafarzadeh, M., and Boult, T. E. (2022). Few-shot class incremental learning leveraging self-supervised features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3900–3910.

Arthur, D. and Vassilvitskii, S. (2007). K-means++ the advantages of careful seeding. In *ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035.

Castro, F. M., Marín-Jiménez, M. J., Guil, N., Schmid, C., and Alahari, K. (2018). End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. (2019). A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*.

Cui, Y., Deng, W., Chen, H., and Liu, L. (2023). Uncertainty-aware distillation for semi-supervised few-shot class-incremental learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Cui, Y., Deng, W., Xu, X., Liu, Z., Liu, Z., Pietikäinen, M., and Liu, L. (2022). Uncertainty-guided semi-supervised few-shot class-incremental learning with knowledge distillation. *IEEE Transactions on Multimedia*.

Cui, Y., Xiong, W., Tavakolian, M., and Liu, L. (2021). Semi-supervised few-shot class-incremental learning. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 1239–1243. IEEE.

Day, W. H. and Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1:7–24.

Dong, S., Hong, X., Tao, X., Chang, X., Wei, X., and Gong, Y. (2021). Few-shot class-incremental learning via relation knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1255–1263.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *IJCV*, page 303–338.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135. PMLR.

Gowda, K. C. and Krishna, G. (1978). Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern recognition*, 10(2):105–112.

Grandvalet, Y. and Bengio, Y. (2004). Semi-supervised learning by entropy minimization. *NeurIPS*.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Hou, S., Pan, X., Loy, C. C., Wang, Z., and Lin, D. (2019). Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839.

Huang, K., Geng, J., Jiang, W., Deng, X., and Xu, Z. (2021). Pseudo-loss confidence metric for semi-supervised few-shot learning. In *ICCV*, pages 8671–8680.

Jia, Y., Abbott, J. T., Austerweil, J. L., Griffiths, T., and Darrell, T. (2013). Visual concept learning: Combining machine vision and bayesian generalization on concept hierarchies. *NeurIPS*, 26.

Kalla, J. and Biswas, S. (2022). S3c: Self-supervised stochastic classifiers for few-shot class-incremental learning. In *ECCV*, pages 432–448. Springer.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Lazarou, M., Stathaki, T., and Avrithis, Y. (2021). Iterative label cleaning for transductive and semi-supervised few-shot learning. In *ICCV*, pages 8751–8760.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S. J., and Yang, Y. (2018). Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Mahalanobis, P. C. (2018). On the generalized distance in statistics. *Sankhyā: The Indian Journal of Statistics, Series A (2008-)*, 80:S1–S7.

Marszalek, M. and Schmid, C. (2007). Semantic hierarchies for visual object recognition. In *CVPR*, pages 1–7.

Martinetz, T., Schulten, K., et al. (1991). A" neural-gas" network learns topologies.

Mensink, T., Verbeek, J., Perronnin, F., and Csurka, G. (2013). Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2624–2637.

Nakata, K., Ng, Y., Miyashita, D., Maki, A., Lin, Y.-C., and Deguchi, J. (2022). Revisiting a knn-based image classification system with high-capacity storage. In *ECCV*, pages 457–474. Springer.

Oreshkin, B., Rodríguez López, P., and Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in neural information processing systems*, 31.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

Peng, C., Zhao, K., Wang, T., Li, M., and Lovell, B. C. (2022). Few-shot class-incremental learning from an open-set perspective. In *ECCV*, pages 382–397. Springer.

Qi, H., Brown, M., and Lowe, D. G. (2018). Low-shot learning with imprinted weights. In *CVPR*, pages 5822–5830.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Ravi, S. and Larochelle, H. (2016). Optimization as a model for few-shot learning. In *International conference on learning representations (ICLR)*.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*.

Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., Keysers, D., and Houlsby, N. (2021). Scaling vision with sparse mixture of experts. *NeurIPS*, 34:8583–8595.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Samek, W., Wiegand, T., and Müller, K.-R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR.

Singh, M., Duval, Q., Alwala, K. V., Fan, H., Aggarwal, V., Adcock, A., Joulin, A., Dollár, P., Feichtenhofer, C., Girshick, R., et al. (2023). The effectiveness of mae pre-pretraining for billion-scale pretraining. *arXiv preprint arXiv:2303.13496*.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. *NeurIPS*.

Srivastava, N. and Salakhutdinov, R. R. (2013). Discriminative transfer learning with tree-based priors. *NeurIPS*, 26.

Sun, L., Wang, M., Zhu, S., and Barbu, A. (2024). A novel framework for online supervised learning with feature selection. *Journal of Nonparametric Statistics*, pages 1–27.

Tao, X., Hong, X., Chang, X., Dong, S., Wei, X., and Gong, Y. (2020). Few-shot class-incremental learning. In *CVPR*, pages 12183–12192.

Tipping, M. E. and Bishop, C. M. (1999a). Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482.

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

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *NeurIPS*.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset.

Wang, B. and Barbu, A. (2022). Scalable learning with incremental probabilistic pca. In *IEEE International Conference on Big Data*, pages 5615–5622.

Wang, Y., Xu, C., Liu, C., Zhang, L., and Fu, Y. (2020). Instance credibility inference for few-shot learning. In *CVPR*, pages 12836–12845.

Yang, B., Lin, M., Liu, B., Fu, M., Liu, C., Ji, R., and Ye, Q. (2021). Learnable expansion-and-compression network for few-shot class-incremental learning. *arXiv preprint arXiv:2104.02281*.

Zhang, C., Cai, Y., Lin, G., and Shen, C. (2020). Deepemd: Differentiable earth mover's distance for few-shot learning. *arXiv preprint arXiv:2003.06777*.

Zhang, C., Song, N., Lin, G., Zheng, Y., Pan, P., and Xu, Y. (2021). Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12455–12464.

Zhou, Z.-H. and Li, M. (2010). Semi-supervised learning by disagreement. *Knowledge and Information Systems*, page 415–439.

Zhu, K., Cao, Y., Zhai, W., Cheng, J., and Zha, Z.-J. (2021). Self-promoted prototype refinement for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6801–6810.

Zweig, A. and Weinshall, D. (2007). Exploiting object hierarchy: Combining models from different category levels. In *ICCV*, pages 1–8. IEEE.

# BIOGRAPHICAL SKETCH

Ke Han received his Bachelor’s degree in Financial Engineering in 2012. In 2015, he earned his M.A. in Statistics from George Washington University. He then pursued his Ph.D. in Statistics at Florida State University, starting it from 2019. Under the supervision of Dr. Adrian Barbu, Ke began his research in machine learning in 2021, focusing on semi-supervised learning and large-scale learning methods to address the challenge of limited labeled data in real-world classification tasks with large output spaces. In 2024, he co-authored a paper with Dr. Barbu titled ”Large-Scale Few-Shot Classification with Semi-Supervised Hierarchical k-Probabilistic PCAs,” which was published in IJCNN 2024.