View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Mountain Scholar (Digital Collections of Colorado and Wyoming)

THESIS

ONE-SHOT LEARNING WITH PRETRAINED CONVOLUTIONAL NEURAL NETWORK

Submitted by

Zhixian Yu

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2019

Master’s Committee:

Advisor: Bruce Draper

Ross Beveridge
Chris Peterson

Copyright by Zhixian Yu 2019

All Rights Reserved

ABSTRACT

ONE-SHOT LEARNING WITH PRETRAINED CONVOLUTIONAL NEURAL NETWORK

Recent progress in convolutional neural networks and deep learning has revolutionized the image classification field, and computers can now classify images with a very high accuracy. However, unlike the human vision system which efficiently recognizes a new object after seeing a similar one, recognizing new classes of images requires a time- and resource-consuming process of retraining a neural network due to several restrictions. Since a pretrained neural network has seen a large amount of training data, it may be generalized to effectively and efficiently recognize new classes considering it may extract patterns from training images. This inspires some research in one-shot learning, which is the process of learning to classify a novel class through one training image from the novel class. One-shot learning can help expand the use of a trained convolutional neural network without costly model retraining. In addition to the practical application of one-shot learning, it is also important to understand how a convolutional neural network supports one-shot learning. More specifically, how does the feature space structure to support one-shot learning? This can potentially help us better understand the mechanisms of convolutional neural networks.

This thesis proposes an approximate nearest neighbor-based method for one-shot learning. This method makes use of the features produced by a pretrained convolutional neural network and builds a proximity forest to classify new classes. The algorithm is tested in two datasets with different scales and achieves reasonable high classification accuracy in both datasets. Furthermore, this thesis tries to understand the feature space to explain the success of our proposed method. A novel tool generalized curvature analysis is used to probe the feature space structure of the convolutional neural network. The results show that the feature space curves around samples with both known classes and unknown in-domain classes, but not around transition samples between classes or out-of-domain samples. In addition, the low curvature of out-of-domain samples is correlated with the inability of a pretrained convolutional neural network to classify out-of-domain classes, indicating that a pretrained model cannot generate useful feature representations for out-of-domain samples.

In summary, this thesis proposes a new method for one-shot learning, and provides insight into understanding the feature space of convolutional neural networks.

ACKNOWLEDGEMENTS

It has been wonderful two years in CSU and there are so many people and things to thank for that there is not even enough space to express my gratitude. CSU is the place where I consider my life is changed completely and without the admission letter from the computer science department, I would be still doing biology experiments.

First, I would like to thank my advisor Dr. Bruce Draper for all his efforts in making me a better computer scientist. Dr. Draper is one of the smartest men I have ever known, and his passion, knowledge and persistence in understanding complex topics deeply inspired me. He is also great at interpreting problems from a very novel and incisive angle. This thesis would never be possible without his help. It was my honor to join the vision lab and such a privilege to have Dr. Draper as my mentor.

I also want to thank Dr. Ross Beveridge and Dr. Chris Peterson for their guidance and input. Meetings with Dr. Beveridge helped me a lot with my research by preventing me from researching in dead ends, and input from Dr. Peterson also improved my understanding about my research.

In addition, I want to thank all my friends in the vision lab: Prady, Dhruva, Rahul, Guru, David and many others. Your company made my research journey much more enjoyable and easier. I wish we had more chances to talk about our researches and other things.

Finally, I would like to thank my wife Kena Shi, who kindly supported my all the way. Not matter whether I was staying up late for homework or getting up early for office hours, she was always there for me. Her emotional support made me go through all the difficulties and pushed me to graduate in time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 Introduction

1.1 Motivation

The development of deep convolutional neural network (CNN) has revolutionized the image classification task. However, the low efficiency of training a CNN begs a better solution to use a pertained CNN model to recognize new classes, which is the goal of this thesis. This thesis proposes an approximate nearest neighbor-based algorithm for recognizing both known and unknown classes. In addition, this thesis tries to probe the structure of the CNN feature space to better understand the feature space and our approximate nearest neighbor algorithm.

Computer vision has a goal of using computers to replicate human vision system in order to either assist humans or to completely relieve humans from some routine tasks. Since the 1960s, the field of computer vision has developed many ground-breaking theories and techniques that have the capability to partially replace some functionality of the human vision system. One particularly important aspect of computer vision is image classification, a rather easy task for humans, yet very difficult for computers until recently. With recent progress in convolutional neural network (CNN) and deep learning (Section 2.1), computers are now able to almost perfectly classify small-scale dataset such as handwriting numeric digits and identify images in a large dataset such as ImageNet in a very high accuracy. The existing academic achievements in this field have inspired further researches that were not possible before. One exciting example is the peer-to-peer communication system built in our lab, where humans can communicate with virtual agents using gestures (Narayana *et al.*, 2018). This system is heavily dependent on the virtual agent's ability to classify hand gestures, achieved by a newly trained CNN model (referred to as HandNet in the following text).

All the recent development and progress mentioned above is dependent on a method called deep convolutional neural network (CNN). With more technical details in Section 2.1.2, CNNs specifically for image recognition usually takes an input of images, which goes through multiple stacked convolutional and pooling layers and fully-connected layers with a softmax function yielding confidence for each class.

The predicted class of the input sample will be the class with highest probability presented by softmax. One can abstract the concept of layers, and then CNN can be considered as consisting of a mapping from the image space to “feature space” and an additional operation on the calculated “features” to get the class label. The following equation mathematically describes this:

$$y_i = g(f(x_i)) \tag{1.1}$$

In this equation, x_i is an image sample, and y_i is the predicted class. f is a mapping from the image space to the feature space, implemented by a series of stacked convolutional and pooling layers and potentially some fully-connected layers. g represents the last fully-connected layer with a softmax function and a argmax() method, determining the predicted classes from the results of $f(x_i)$. Although the exact architecture may differ, the idea of stacking multiple layers of convolutional and pooling operations is essential in all the deep CNN implementations.

Despite the practical success of CNN, it is far from a precise replication of the human vision system. The human vision system can recognize new objects after seeing a similar one, which is both effective and efficient. However, a pretrained CNN model can only be used to recognize a fixed number of classes. Intuitively, the architecture of the CNN model can be modified to recognize new classes. However, retraining a CNN model requires large amounts of training data and significantly long training time. The reason for such a high training cost lies in the high complexity of deep CNN models which demand both a large input size to prevent overfitting and long training time for optimized parameter sets. Thus, it is not an efficient way to change and retrain a CNN model to recognize new classes. Considering a pretrained CNN model has seen a large amount of training images, it may be able to extract useful common patterns and may be generalized to recognize other classes. Therefore, it may be possible to use a pretrained CNN model and its generalized information to recognize new classes, which provides a motivation for this thesis. The contribution of this thesis is to propose a method to efficiently and effectively recognize new classes with a pretrained CNN model without retraining it and to understand the mechanisms behind it.

1.2 Thesis Contributions

There are mainly two contributions in this thesis: proposing a method for one-shot learning based on CNN and ANN algorithm and a tentative examination of the structure of CNN feature space through generalized curvature analysis (GCA). The ANN algorithm in the form of a proximity forest is introduced in Section 2.3. As mentioned above, the first step of classification by CNN can be viewed as a mapping from the image space to the feature space. Since a pretrained CNN model has seen a large amount of training images and can recognize known classes in a high accuracy, it may extract useful patterns from input samples and generate meaningful feature representations that can be used for easily recognizing different classes. The ANN algorithm is first used to replace softmax function to classify known classes. This thesis finds that, compared to a standard "fully-connected layer + softmax" classifier, the ANN algorithm can better classify hand gestures, which belong to a smaller scale dataset, and has a comparable performance in classifying images from a very large-scale dataset, the ImageNet.

In addition to using ANN algorithm to classify known classes, the ANN algorithm is also tested for one-shot learning, which is the process of learning a new class through one training sample from the new class. The assumption is that a pretrained CNN model can also generate good feature representations for new classes since it has seen enough training images. This thesis finds that the ANN algorithm can classify samples from novel classes with a reasonably high accuracy for both the small-scale hand gesture dataset and large-scale ImageNet dataset. Therefore, the first major contribution is the proposal of using ANN algorithm to effectively classify known classes and to perform one-shot learning.

Following the practical success of replacing softmax with ANN for classifying both known and unknown classes, this thesis tries to understand the mechanisms. The hypothesis is that the CNN feature space curves around trained classes and new classes so that the samples from the same class seem clustered together. A novel tool, generalized curvature analysis (GCA), is used to probe the structure of the feature space. The GCA is introduced in Section 2.4 and operates on the CNN features of a set of images. Since each sample is mapped in the feature space as a point, connecting different points from different samples creates a curve in the high-dimensional feature space. The curvature of the resulting curve can be calculated to gain some insight into the structure of the feature space. This thesis finds that the curvature is generally lower during transitions between gesture classes, but higher in the middle of a gesture class, indicating the feature space curves around samples from each class, but not samples within transitions. This pattern is also true for new classes, indicating that the CNN feature space also curves around samples from new classes.

Both known and unknown classes share the same pattern with original training classes, so they can be considered as “in-domain” samples. It is not known how the feature space structures around out-of-domain samples, which presumably look very differently from the training samples. Therefore, this thesis tries to analyze the curvature with out-of-domain samples to gain some insight into the feature space structure around those samples. This thesis finds that out-of-domain samples have lower curvature compared with in-domain samples, indicating that the CNN feature space does not curve around out-of-domain classes. This also suggests that a pretrained CNN model could not generate meaningful representations for out-of-domain samples, which further suggests that these samples could not be effectively learned in a one-shot manner. To test the one-shot learning capabilities of the out-of-domain samples, the capabilities are compared with those of the in-domain unknown samples. This thesis finds that the out-of-domain classes cannot be distinguished or learned using our ANN algorithm, further confirming the idea that a pretrained CNN model cannot generate useful feature representations for out-of-domain classes. Therefore, the second major contribution is that the CNN feature space is indicated to curve around known and new classes, but not samples during class transition or out-of-domain samples. In addition, the CNN feature space may not be able to generate useful feature representation for out-of-domain classes, which cannot be effectively learned in a one-shot manner.

In summary, this thesis proposes an effective method, ANN algorithm with CNN features, for classifying known classes and one-shot learning. It also uses GCA to probe the structure of the feature space with results indicating that the CNN feature space curves around known and unknown in-domain classes, but not samples during transitions between classes or out-of-domain classes. This provides an initial understanding of the structure of the CNN feature space.

1.3 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 introduces necessary background and literature review on the following topics: CNN and one-shot learning (Sections 2.1 and 2.2), ANN (Section 2.3) and GCA (Section 2.4). Chapter 3 describes the methods and results for one-shot learning using our proximity forest-based ANN algorithm. Chapter 4 talks about theoretical analysis on one-shot learning and CNN feature spaces based on the results from GCA. Chapter 5 discusses the conclusions and future directions.

Chapter 2 Background

2.1 Convolutional neural network (CNN)

2.1.1 Brief history of image classification and CNN

As a core task of computer vision, image classification focuses on assigning categorical labels for images. It is not an easy task for an automated computer system, although it is rather routine for humans whose daily activities heavily depends on classification. Humans can quickly recognize an object if they were taught with similar things before. For example, a person can easily recognize an object in an image as a cup because it has all features of a cup as previously seen. Following with similar ideas, one may train programs with pictures of cups, and hope they can recognize new cups on their own. However, it was far from a simple task until recent developments in deep learning, which is a data-driven technique and can train the feature extractor and classifier at the same time. Among different deep learning architectures, the deep convolutional neural network (CNN) has become the dominant architecture for solving image classification and other problems (Lecun, Bengio and Hinton, 2015).

The first instance of a CNN was first proposed and implemented by Lecun *et al.* (LeCun *et al.*, 1989; Cun *et al.*, 2000). The CNN models were originally successfully applied in postal hand-written digits recognition. However, the development of deep CNNs was hindered until later adoption of several valuable techniques: GPU-accelerated gradient descent training technique (Chellapilla *et al.*, 2006), application of a rectified linear unit (ReLU) (Nair and Hinton, 2010), and the emergence of super large datasets such as the ImageNet (Russakovsky *et al.*, 2015) and the human motion dataset (Kuehne *et al.*, 2011). GPU acceleration and use of ReLU activation function significantly reduce training time, and the ReLU function also mitigates the vanishing gradient problem (Glorot, Bordes and Bengio, 2011). In addition, using a large dataset reduces the chances of overfitting, a real risk given the extremely high number of weights to be trained.

In 2012, Krizhevsky et al. developed a ground-breaking deep CNN model, the AlexNet, which won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) (Krizhevsky, Sutskever and

Hinton, 2012). Since then, multiple new architectures have been developed with much better classification capabilities (two of them are introduced in Section 2.2). In addition, many other fields, such as natural language processing, have also adapted the CNN models to achieve their goals (Lecun, Bengio and Hinton, 2015). The next two sections will discuss common CNN architectures and potential problems with them. In the following text, CNN will refer to deep CNN unless mentioned otherwise.

2.1.2 Common CNN architecture

The common architecture of a CNN model is shown in Figure 2.1. An image classification CNN model usually takes an image as input, which goes through several layers of convolution and pooling operations. The convolutional layer is a core building block in a CNN model with a primary goal of extracting features. Each layer contains a set of kernels which have a small receptive field but extend through the full depth of the input. The first convolutional layer detects low-level features such as edges, and the higher layers learn more abstract features (Gu *et al.*, 2015). For each layer, ReLU activation function is generally used. The most typical pooling operation is max pooling (Boureau, Ponce and LeCun, 2010), whose objective is to down-sample the input. The max pooling process takes values from a small receptive field as input and outputs the largest value in that field. This process can abstract the input representation and reduce computation cost by reducing the number of parameters in the following layer. By stacking several convolutional and pooling layers, one may gradually extract higher-level feature representations.

The result of the last pooling layer is flattened to a one-dimensional vector, followed by one or more fully-connected layers. The last fully-connected layer usually has as many outputs as the number of object classes to be recognized. The unit with the highest activation signifies the object being recognized. To accentuate the winning unit, a softmax function is applied to the output of the last fully-connected layer yielding pseudo-probabilities for each class.

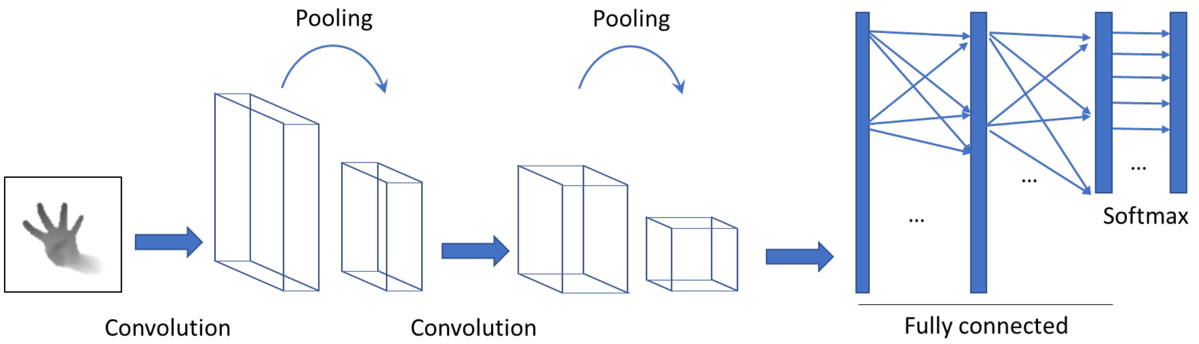

Figure 2.1. An illustrative figure showing a common architecture of a deep CNN model.

Here precisely is how a softmax function is defined. For a particular class j from all K classes, the softmax function is:

$$p_j = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}} \tag{2.1}$$

where for $j = 1, \dots, K$,

$$z_j = \boldsymbol{x}^T \boldsymbol{w}_j \tag{2.2}$$

where $\boldsymbol{x}$ is the output of the last fully-connected layer (i.e., the input for the softmax function), and $\boldsymbol{w}$ is the weight matrix. From equation (2.1), each probability from the softmax function has a range of (0,1), and the sum of all probabilities for all classes is 1. In addition, it can only be used to choose among K classes as defined at the beginning of the training. In other words, the softmax will always generate K probabilities after training, and adding a new class requires modification of the CNN architecture.

2.1.3 One-shot learning

Although CNNs achieve state-of-art performances in many fields including image classification, they are not comparable to human vision in terms of efficiency. More specifically, given an object that was never seen before, humans can quickly learn the patterns from even one single sample and effectively identify similar ones. However intuitively, CNNs seem to lack this ability since using a CNN model to classify images requires significant resources of both training images and time. For example, reported training time of the Alex net was 5 to 6 days on two GTX 580 3GB GPUs (Krizhevsky, Sutskever and

Hinton, 2012), and training a ResNet-50 model with 8 Tesla P100 GPUs requires 29 hours (Ojima, Yamaguchi and Taya, 2018). Therefore, the intuitive way of introducing a new class into the classification problem by modifying the structure and retraining the model is not comparable with human vision in terms of efficiency.

The goal of better approaching the efficiency of human vision has inspired researches in one-shot learning, which is the process of learning to classify a novel class through one training image from the class (Vinyals *et al.*, 2016). This can be extended to few-shot learning, which learns based on a few new samples (Sachin Ravi, 2017). The assumption under one-shot learning is that since the existing models have already seen enough training images, it may be possible to generalize the patterns. An argument is that the convolutional layers, especially the bottom convolutional layers, learn low-level feature patterns that are general to all images (Krizhevsky, Sutskever and Hinton, 2012). In addition, it was shown that the features extracted from the activation of a CNN can be repurposed to novel generic tasks (Cui *et al.*, 2017). In other words, the feature extraction process from a trained CNN model can be used to extract meaningful features in a different task. Therefore, trained models may have the potential to recognize new classes even though there are only a few samples available.

Recent developments in one-shot learning is summarized next, and the existing implementations of one-shot learning can be categorized to the following groups.

- Naïve methods. The most straightforward way to perform one-shot learning is through a complete nearest neighbor search on pixel distance defined by correlation, and it is usually used as a baseline for studies (Vinyals *et al.*, 2016). However, it is impractical in most cases to perform a full nearest neighbor search in a large dataset such as ImageNet dataset which contains over 1 million samples for 50,000 testing samples. Consequently, most results report on a subset of ImageNet where 100 classes are randomly sampled. The dataset generated by randomly sampling 100 classes from ImageNet is called miniImageNet.

- Non-deep learning-based methods. Fei-Fei Li *et al.* (Fei-Fei, Fergus and Perona, 2006; Strassburger, 2006) developed a Bayesian framework to perform one-shot learning under the premise that previous learned “generic” features from unrelated classes can help identify the new class, and they achieved an error rate of 8-22% on 4 classes (3 training and 1 testing). In a different paper (Lake *et al.*, 2011), a character recognition problem was addressed through first creating a generative hierarchical Bayesian model to analyze strokes from characters. The prior knowledge on known character strokes can be used to help infer strokes in new characters and recognize them with a reported accuracy of 63.7 % on 20 classes (19 training and 1 testing). This model was further improved to achieve an accuracy of 95.2% (Lake, Salakhutdinov and Tenenbaum, 2013). Non-deep learning-based methods were also used for one-shot gesture recognition. For example, Konečný *et al.* (Konečný and Hagara, 2013) used Histograms of Oriented Gradients (HOG) and Histogram of Optical Flow (HOF) features for one-shot gesture recognition.

- Nearest neighbor directly on CNN features. The idea behind this approach is that extracted CNN features are good representations of images even for classes that were not trained on. The nearest neighbor search can be performed with the feature vectors from Inception models and achieve an accuracy of 36.6% on mini-ImageNet with 100 randomly sampled classes (Vinyals *et al.*, 2016).

- Modified pre-trained CNN models. Since softmax is the major block in classifying new classes, modifying the softmax may directly help. Burgess *et al.* (Burgess, Lloyd and Ghahramani, 2017) proposed a Bayesian method to update a pretrained CNN model. They modeled the weights prior to softmax function as a multivariate Gaussian and predicted new weights for a new class based on prior knowledge on the existing weights.

- CNN models specifically designed for one-shot learning. These adapted CNN models do not have a standard CNN architecture with softmax classification, and they are not used for classifying the original dataset although they can be. Vinyals *et al.* (Vinyals *et al.*, 2016) proposed a model called Matching Networks which maps a small labelled support set and an unlabelled example to its label. Their model can achieve an accuracy of 46.6% in miniImageNet on 5 testing classes. A similar idea is used by a new model called Prototypical Networks which uses a small number of samples for each known class as prototypes, and learns a metric space where the classification can be performed by computing distances to prototype representations of each class (Snell, Swersky and Zemel, 2017). The Prototypical Networks can achieve an accuracy of 49.42% in miniImageNet on 5 testing classes. A Generative Adversarial Networks (GAN) based model achieved an even higher accuracy of 55.2%, which to our knowledge is the state-of-art result (Mehrotra and Dukkipati, 2017). The GAN-based model modified the discriminator and replaced it with a pairwise network for one-shot learning. For completeness, a LSTM-based meta-learner model has an accuracy of 43.44% in miniImagenet (Sachin Ravi, 2017), and a memory-augmented neural network has 36.4% accuracy in the Omniglot dataset (Santoro *et al.*, 2016) .

In this thesis, I will propose a new method of using an approximate nearest neighbor algorithm directly with CNN features. The following sections will provide necessary technical background for understanding the experiments.

2.2 Examples of Deep CNN Models

This section discusses two commonly used CNN models which are later used in this thesis.

2.2.1 ResNet-50 and HandNet

ResNet was first developed by Microsoft Research, and won the ILSVRC2015 with a top-5 error rate of 3.57% (He *et al.*, 2015). It was implemented mainly to solve the saturated performance problem with more convolutional layers stacked. This is mainly caused by the vanishing gradient problem. The core idea of ResNet is the "identity shortcut connection", which introduces a shortcut between two layers and forces the network to learn the residual between the input and output instead of the direct mapping. One residual module is shown in Figure 2.2. This architecture allowed the authors to construct arbitrarily deep networks to improve performance.

Another important modification in the ResNet-50 architecture is that there are no full-connected layers except for the one prior to the softmax function. The output of the last convolutional layer goes through an average pooling operation, which is followed by a fully-connected layer and softmax.

Inspired by the ResNet-50 architecture, our lab (Narayana *et al.*, 2018) adapted it to recognize hand gestures, and used it in a human-computer interaction prototype. Differently from the ResNet-50

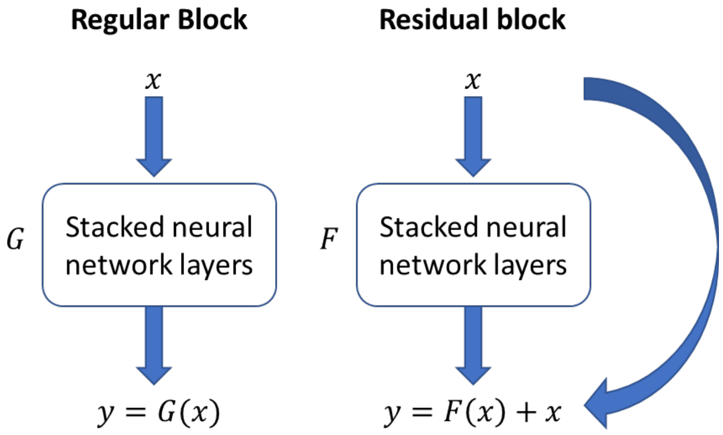

Figure 2.2. Comparison between a regular block in common CNN and a residual block in ResNet. In a residual block, the residual $F(x)$ between the output and the input is learned.

used to classify ImageNet, the output of the convolutional layers has a dimension of 1024, and can be used to generate 32 labels including 30 gestures with additional "blank" and "other" classes. The HandNet was trained with 28, 506 samples of right hands, and has a validation accuracy of 82% (Wang *et al.*, 2017). This thesis used the HandNet as one of its CNN models.

2.2.2 Inception v4

The Inception v4, developed by Google Research, has a top-5 error rate of 4.2% on ILSVRC2012 (Szegedy *et al.*, 2016). It begins with Inception v1 with the idea of stacking convolutional layers not only vertically but also horizontally (Kim *et al.*, 2016). The Inception v1 model is built on modules which perform convolutions with different kernel sizes on the same input and concatenate the results together. In addition, auxiliary classifiers were introduced to diminish the vanishing gradient problem. Later Inception models were improved by introducing 1x1 convolutional masks (adapted from ResNet), reducing larger convolutional kernel to ones with smaller sizes, adding more convolutional filters in one module, making modules more unified, etc. One type of Inception v4 module is shown in Figure 2.3. In this thesis, I used Inception v4 trained on ILSVRC2012, available as a pretrained tensorflow model (https://github.com/tensorflow/models/tree/master/research/slim).

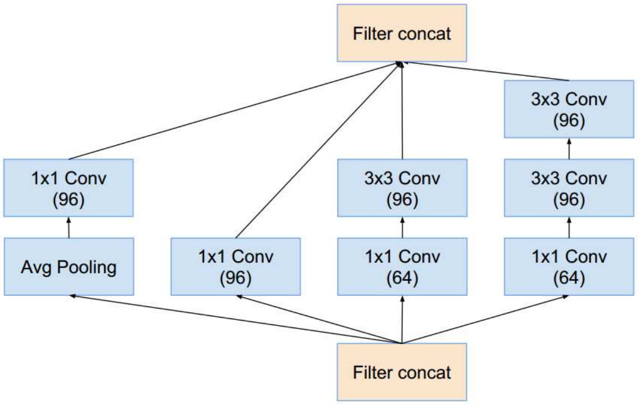

Figure 2.3. An inception block (Inception-A block). Notice the use of 1x1 convolutional layer adapted from ResNet (Szegedy *et al.*, 2016).

2.3 Proximity Forest-Based Approximate Nearest Neighbor Algorithm

This thesis uses an approximate nearest neighbor (ANN) algorithm first proposed and used by O'Hara *et al.* (O'Hara and Draper, 2012, 2013). Briefly, this ANN algorithm is based on a proximity forest which consists of multiple randomized metric trees. During construction of a metric tree, samples are added to an initially empty node until the node reaches a certain size and needs to be split. One sample in the node is chosen as a pivot, and the distance between the pivot and every other sample is computed to split the samples with lower distances to the left node and those with higher distances to the right node. This process is repeated until a complete metric tree is built. Prior work has demonstrated the value of using multiple trees, hence the randomly built trees constitute a forest. This method is applied on CNN features for one-shot learning.

2.4 Generalized Curvature Analysis (GCA)

After the practical use of one-shot learning, this thesis tries to understand the mechanisms by examining the feature space. A novel method GCA is used to probe the structure of CNN feature space. The mathematical argument for GCA was given by (Álvarez-Vizoso *et al.*, 2015). They proved that for a parametric curve γ at a particular point $\gamma(t)$, the Frenet-Serret frame and the local singular vectors agree as the ϵ-ball used for computing the singular vectors gets significantly small, in which case the curvature function can be expressed as a fixed multiple of a ratio of local singular values. More precisely, the ith curvature at $\gamma(t)$ is given by

$$\kappa_{i-1}(t) = \sqrt{a_{i-1}} \frac{\sigma_i(t)}{\sigma_1(t)\sigma_{i-1}(t)} \tag{2.3}$$

where

$$a_{i-1} = \frac{4i^2 - 1}{3} \left(\frac{i}{i + (-1)^i} \right)^2 \tag{2.4}$$

and $\sigma_i(t)$ is the ith local singular values at $\gamma(t)$ for the ϵ-ball.

One real application of the GCA is motion segmentations (Arn *et al.*, 2018). In this study, human skeleton coordinates acquired by Kinect v2 were considered as points in a 75-dimensional space, and a sequence of these skeletons forms a curve in the space. GCA was then used to segment pose streams into motions and transitions without knowing the set of possible motions in advance. This thesis applies the GCA to CNN features to reveal the structure in the feature space.

2.5 Summary

Deep CNN models have gained strong attention because they have leading advantages in image classification. However, training a CNN model requires significant resources. This thesis focuses on finding a new method for one-shot learning without constructing task-specific models. One-shot learning is tested on ResNet-50 based HandNet and Inception v4 models using proximity-forest based ANN algorithm. This thesis further analyzes the structure of CNN feature space using GCA and discovers a potential relationship between curvature and one-shot learning capabilities.

Chapter 3 One-shot learning

Image classification tasks can often be performed with high accuracy thanks to the development of deep convolutional neural network (CNN). However, training a deep CNN model requires a significant amount of data and time. This chapter focuses on using pre-trained CNN models for one-shot learning of untrained classes while minimizing training data size and time. As mentioned in Chapter 2, one-shot learning is the process of learning to classify a novel class based on one training image from the novel class.

This chapter presents 5 experiments to illustrate our proposed one-shot learning algorithm. Section 3.2.1 compares different distance metrics and the negative of Pearson's correlation on CNN features is chosen as the distance metric. To test the efficacy of our proximity forest-based approximate nearest neighbor (ANN) algorithm, two datasets (the hand gesture dataset and ImageNet dataset) are used. Section 3.2.2 uses the ANN algorithm to classify known classes by replacing the softmax classifier for the hand gesture dataset. Similar experiments are done in Section 3.2.3 for ImageNet dataset. Sections 3.2.4 and 3.2.5 discuss the results of one-shot learning for the hand gesture dataset and ImageNet dataset, respectively.

3.1 Methods

To test our one-shot learning algorithm, two different CNN models and their associated datasets are used. A previously published ANN algorithm, as described in Chapter 2, is used to perform one-shot learning. Sections 3.1.1 and 3.1.2 introduce the two different datasets and models, respectively. The exact ANN parameters are listed in Section 3.1.3.

3.1.1 Datasets

This thesis tests the ANN algorithm with two datasets: the hand gestures extracted from the EGGNOG dataset (Wang *et al.*, 2017) and the ImageNet dataset (Russakovsky *et al.*, 2015).

The EGGNOG dataset captures aspects of communication through gestures and features over 7 hours of footage over 360 videos. This thesis does not use the whole videos, but the hand gestures extracted from them. More specifically, based on the depth images given by Microsoft Kinect v2, hand depth images are center-cropped around the palm center, based on the coordinate provided by the Kinect skeleton analysis. To normalize the raw hand depth image, the center pixel value of a hand depth image is first subtracted from every pixel. Because of the Kinect v2 factory settings, most of the pixels in the resulting images have a value between -150 and 150. Then every pixel value of the resulting image is divided by 150 to be within the range of [-1,1]. Any pixel with a value lower than -1 is set to -1, and that higher than 1 is set to 1. The final images are resized to (128, 128) and used as the input for CNN models. Both training and testing images are captured using the same method. Figure 3.1 shows 4 representative hand gestures used for training a CNN model.

The ImageNet dataset contains 14,197,122 images with over 20,000 categories, although the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) only uses a trimmed set of 1,000 classes with over 1 million images. Most well-recognized CNN models such as VGGNet, ResNet and Inception use the 1,000-class ImageNet (Simonyan and Zisserman, 2014; He *et al.*, 2015; Szegedy *et al.*, 2016). Since the first deep CNN model AlexNet used the ILSVRC2012 dataset (Krizhevsky, Sutskever and Hinton, 2012), many following studies reported accuracies on the ILSVRC2012 dataset. Consistent with prior work, this thesis uses the ILSVRC2012 dataset, which contains over 1 million training images and 50,000 validation images (50 images/class with 1,000 classes).

3.1.2 CNN models and features

This thesis uses two different pre-trained CNN models, the HandNet (Narayana *et al.*, 2018) and Inception v4 (Szegedy *et al.*, 2016), for the hand gestures and ImageNet dataset, respectively. The

HandNet, modified based on the architecture of ResNet-50, was trained by a previous lab member Pradyumna Narayana. It takes a normalized hand gesture image as described above as input and can be used to classify 30 natural hand gestures, 4 of which are shown in Figure 3.1. As shown in Figure 2.1, the vector prior of the last fully-connected layer and softmax is considered as the feature vector for each input. The dimension of a feature vector from a HandNet is 1024.

The pre-trained Inception v4 model (Szegedy *et al.*, 2016) is available from the tensorflow slim package (https://github.com/tensorflow/models/tree/master/research/slim). Similar to the HandNet, the vector prior of the last fully-connected layer and softmax is used as the feature vector for input ImageNet images. The dimension of a feature vector from Inception v4 is 1536.

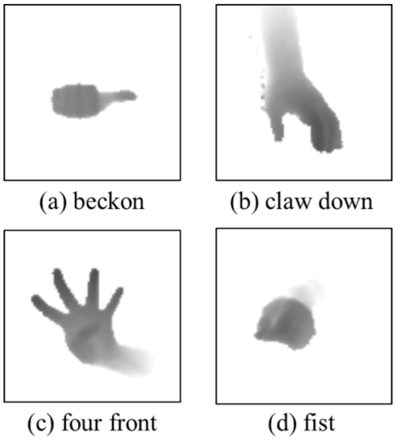

Figure 3.1. Sample hand depth images showing 4 different hand gestures.

3.1.3 Parameters of proximity forest

The implementation of ANN algorithm is based on a previously published proximity forest algorithm (O'Hara and Draper, 2012, 2013). As described in Chapter 2, there are several parameters that can be tuned to build a proximity forest. The maximum number of items in a node determines the capacity of a node and therefore indirectly controls the depth of a proximity tree. In an extreme situation, if the maximum number of items in a node equals the total number of samples, the ANN algorithm will be equal to an exact nearest neighbor algorithm. The number of trees in the forest affects the final classification accuracy. Theoretically, more trees result in higher accuracy but lead to diminishing returns and increased run time. The distance metric is discussed in Section 3.2.1. Several splitting methods were mentioned in the original paper (O'Hara and Draper, 2013), and the median splitting is chosen for simplicity. The ensemble strategy is the final step of the ANN algorithm. Theoretically the majority vote provides the most stable results, although it costs more to compute. Finally, to increase efficiency, all training samples are not used. The parameters for constructing a proximity forest are summarized in Table 3.1 and Table 3.2.

Table 3.1. Proximity forest parameters for ANN-based classification of hand gestures.

Parameters	Values
Maximum number of items in a node	21
Number of trees	25
Distance metric	Negative of Pearson's correlation
Splitting method	Median splitting
Ensemble strategy	Find the closest samples from each tree, report the label of the closest sample from all found samples
Number of reference images	1,000 images/class; totally 31,000 images

Table 3.2. Proximity forest parameters for ANN-based classification of ImageNet images.

Parameters	Value
Maximum number of items in a node	151
Number of trees	50 or 60
Distance metric	Negative of Pearson's correlation
Splitting method	Median splitting
Ensemble strategy	Majority vote

Number of reference images	200 images/class or all training images

3.2 Results

3.2.1 Experiment 1: Selecting a distance measure

The first step of building a proximity forest is to select a distance measure. Since an approximate nearest neighbor (ANN) algorithm is used, the features from the same class will ideally have significantly smaller distances so that they cluster together, and those from different classes will have significantly larger distances so that they become well separated. Three common distances measures are considered: Pearson's correlation, Euclidean distance and chordal distance. Each measure is discussed in depth below.

To calculate the Pearson's correlation, each feature vector x (input of the last fully-connected layer as mentioned in Section 2.2.1) is first standardized by subtracting the mean and dividing the standard deviation as shown in equation (3.1).

$$x' = \frac{x - \bar{x}}{\sigma} \tag{3.1}$$

where the standard deviation $\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$, and n is the dimension of feature vector (1024 for HandNet raw images and features, and 1536 for Inception v4 features). The distance measure of a sample pair $d(x, y)$ is then computed using equation (3.2).

$$d(x, y) = -\frac{1}{n-1}\sum_{i=1}^{n} x_i' y_i' \tag{3.2}$$

Figure 3.2 shows the distribution of distance measures from intra-class and inter-class pairs for four different hand gestures: beckon, claw down, four front and fist. As is shown in Figure 3.2, the negative of Pearson's correlation distance measure for intra-class samples are obviously smaller than those for inter-class samples. This indicates our distance measure can cluster samples from the same class together, and at the same time separate the samples from different classes. This is ideal for an ANN algorithm which requires samples with the same label mapped closer in space. This indicates the negative of Pearson's correlation can be a good candidate for the ANN algorithm.

We also consider the Euclidean distance. However, since the Euclidean distance and Pearson's correlation describe the same relative distance relationship between unit vectors, they serve the same purpose in building a proximity forest. At the same time, the Pearson's correlation can be efficiently computed as a dot product between two unit vectors, so we choose the negative of Pearson's correlation rather than Euclidean distance. For completeness, the results for Euclidean distance are shown in 0.

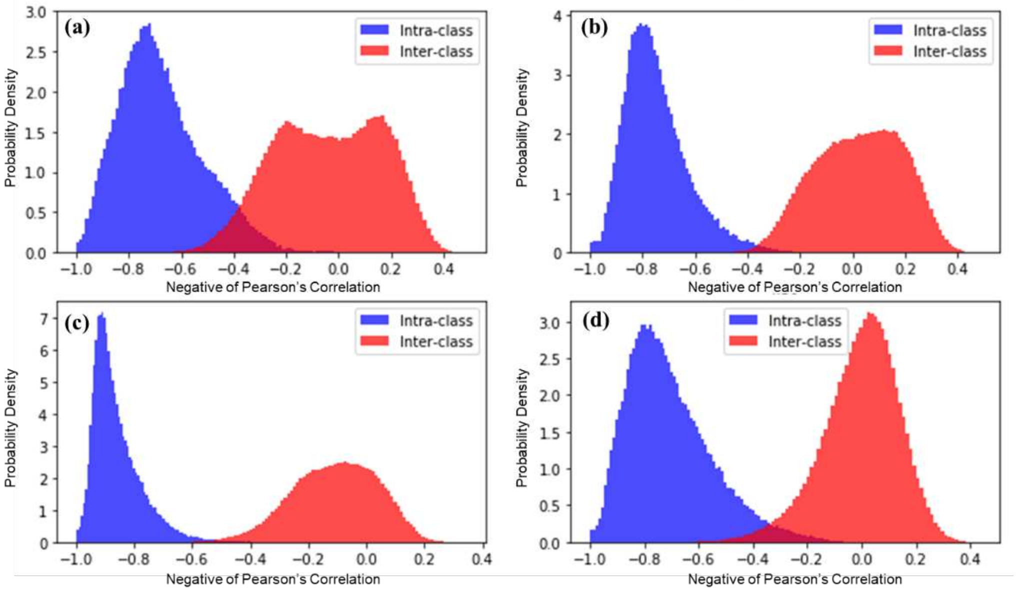

Figure 3.2. Histograms showing the inter-class and inter-class sample distances computed using Pearson's correlation on CNN features. The hand gesture classes are: **(a)** beckon; **(b)** claw down; **(c)** four front; **(d)** fist.

The chordal distance is similar to the one proposed in the original reference (O'Hara and Draper, 2012). Briefly, the input, instead of the output, of the last average pooling layer is used as feature representation for each sample (see Section 2.2.1 for detailed architecture of the ResNet-50 and HandNet). Therefore, each feature representation has a dimension of 1024x64 because of the architecture of the HandNet. Let T_i be the feature representation for an input image i. Using QR Factorization, we have:

$$T_i = Q_i R_i \tag{3.3}$$

where Q_i is an orthogonal basis for T_i. The cosine distance between two feature representations T_i and T_j can be computed using the principle angles Θ between the subspace spanned by Q_i and Q_j:

$$\begin{gathered} Q_i^T Q_j = U\Sigma V^T \\ cos\Theta = diag(\Sigma) \end{gathered} \tag{3.4}$$

The chordal distance between two feature vectors $d(T_i, T_j)$ is defined as the L2 norm of the component-wise sine function applied to Θ:

$$d(T_i, T_j) = \|sin\Theta\|_2 \tag{3.5}$$

Figure 3.3 shows the distribution of chordal distance measures from intra-class and inter-class pairs for four different hand gestures: beckon, claw down, four front and fist. As is shown, the chordal distance measure for intra-class samples do not have significantly different distributions from those for inter-class samples. This is particularly true for the fist class, for which the intra-class group and inter-class group have very similar distributions (Figure 3.3 (b)). This indicates that using chordal distance measure may not effectively cluster samples from the same class or separate the samples from different classes, and thus the ANN algorithm is not expected to highly correctly assign labels for samples.

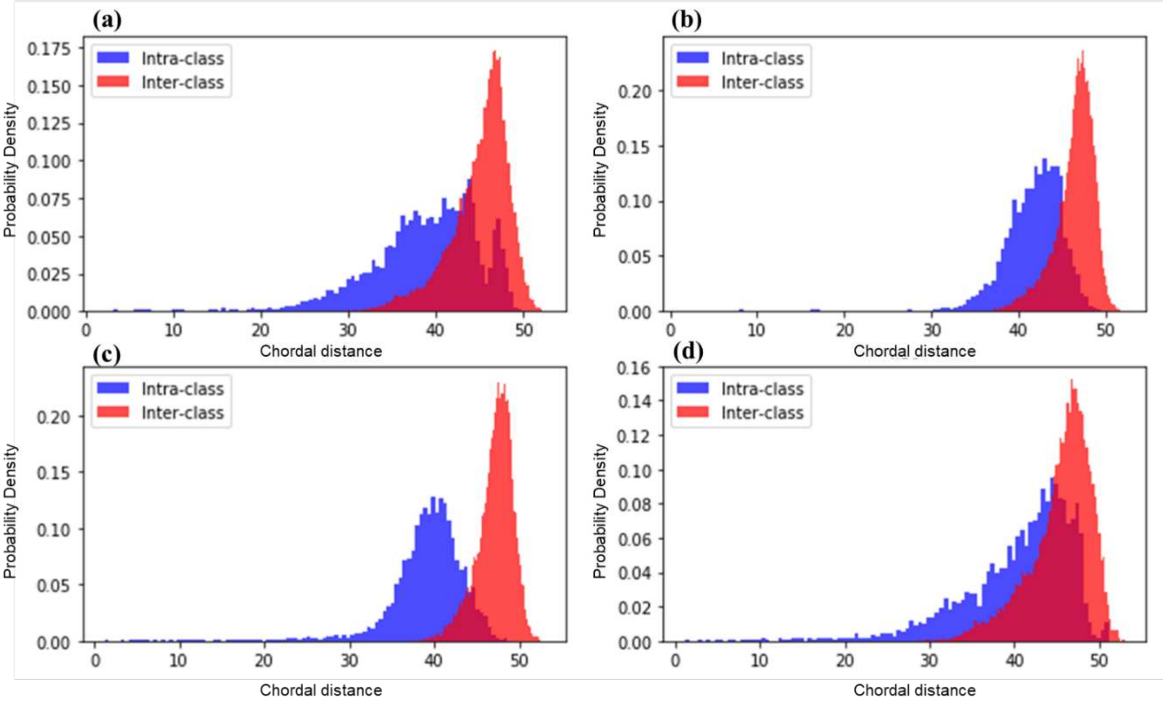

Figure 3.3. Histograms showing the inter-class and inter-class sample distances computed using chordal distance. The hand gesture classes are: **(a)** beckon; **(b)** claw down; **(c)** four front; **(d)** fist.

In summary, three distance measures are compared in this section. Based on the understanding on the mechanism of ANN algorithm and the intuitive interpretation of the distance distribution, the negative of Pearson's correlation is chosen as the distance measure because the intra-class distances are small and the inter-class distances are large. In other words, it is expected easier to separate the samples from different classes using the negative of Pearson's correlation rather than the chordal distance with the ANN algorithm. The Euclidean distance is not chosen because of the increased computation cost. The negative of Pearson's correlation is used as the distance measure in all following ANN algorithm implementations.

3.2.2 Experiment 2: Using CNN features and ANN algorithm for gesture classification

After selecting the distance measure, the next step is to determine if the proximity-forest based ANN algorithm with CNN features can be used to correctly label samples. As a proof of concept, the

ANN algorithm is first tested on known hand gesture classes instead of directly for one-shot learning to determine the efficacy of replacing softmax function with the ANN algorithm to classify known gestures.

The HandNet with softmax gives 82.5% testing accuracy on known hand gesture testing samples, which contain 31,065 hand gesture images belonging to 30 gesture classes (Table 3.3). To determine the efficacy of using ANN algorithm to classify these testing samples, a proximity forest is first constructed using the CNN features from the original training samples based on parameters mentioned in Table 3.1. The CNN features of the testing samples are extracted through the HandNet, and the label of each testing feature is determined by searching the constructed proximity forest. As shown in Table 3.3, using ANN algorithm increased the testing accuracy by 2.77%.

Table 3.3. Testing accuracy of classifying hand gestures using two classifiers.

Classifier	Accuracy
Fully-connected layer with softmax	82.5%
ANN	85.27%

This result first indicates that the negative of Pearson's correlation is a practically good distance measure for the ANN algorithm with CNN features, which is consistent with the finding in Section 3.2.1. In addition, the result shows that our ANN algorithm can be used for effectively classifying hand gestures. In fact, the ANN algorithm performs better than the softmax function, which was originally used to train the HandNet. Although it is only 2.77% higher, this is a surprising result because a previous work reported 0.1% decrease in classification accuracy when replacing softmax with a 75-nearest neighbor classifier in MNIST (Papernot and McDaniel, 2018), and a different paper also reported decreased accuracy with a 1-nearest neighbor search in miniImageNet (Vinyals *et al.*, 2016). This increased accuracy may be specific to our HandNet and gesture dataset, but it suggests that the HandNet features are good representations of gesture images in terms of spatial organizations of the feature vectors: since the ANN algorithm works well when samples from the same class are near each other in the sample space, CNN features from the same gesture class may cluster together, and those from different classes may be further apart.

3.2.3 Experiment 3: Using CNN features and ANN algorithm for ImageNet classification

Our HandNet is small-scale in terms of the number of recognizable classes and the number of training images used. To test whether our ANN algorithm is effective in classifying large-scale dataset, CNN features from all ILSVRC2012 training images are extracted using the Inception v4 model and used to build a proximity forest, which is used to classify 50,000 testing samples. This experiment does not involve new classes and is used as a proof of concept.

Table 3.4 compares the testing accuracies of classifying ILSVRC2012 validation images using original softmax and our ANN algorithm. The testing accuracy using ANN algorithm is only marginally (1.58%) lower than the softmax. Compared with the results from gesture dataset (Table 3.3) and previous literature (Vinyals *et al.*, 2016; Papernot and McDaniel, 2018), the efficacy of replacing softmax function with ANN algorithm is not optimal. However, this is probably because the ImageNet is a very large-scale dataset, and the Inception v4 is a more complex model. There are 1000 classes, over a million features in the forest to compare with, and 50,000 testing samples. In addition, the feature vector size is 1024 for the HandNet, but 1536 for the Inception v4, which can also potentially increase the complexity. Therefore, 1.58% decrease in classification accuracy may not be a dramatic effect.

Table 3.4. Testing accuracy of classifying ILSVRC2012 validation images using two classifiers.

Classifier	Accuracy
Fully-connected layer with softmax	80.18%
ANN	78.6%

The fact that ANN algorithm can achieve a comparable classification accuracy with softmax function indicates that the features extracted using Inception v4 are also relatively good feature representations of their original images similar to the features from the HandNet. In addition, this is the first report on using nearest neighbor/ANN algorithm on the full ImageNet to my best knowledge, showing that the ANN algorithm is effective in classifying a large-scale dataset.

The results from Section 3.2.2 and Section 3.2.3 together indicate that CNN features from the same known class tend to cluster together and those from different classes further apart. In addition, a proximity forest-based ANN algorithm is effective in classifying images based on their feature representations for known classes. This lays the foundation for using pre-trained CNN models and ANN algorithm for one-shot learning.

3.2.4 Experiment 4: One-shot learning of hand gestures

The previous two experiments demonstrate the performance of using CNN features and ANN algorithm by classifying images with known classes. Since the training of a CNN model requires many training images, it is possible that CNN models might be generalized to recognize new classes as well. In other words, the CNN features from new-class images may also be good representations. However, the softmax classifier cannot be directly used for generating probabilities for a new class, giving ANN algorithms an advantage in classifying new classes.

The ANN algorithm is first tested on the HandNet with a new class, American Sign Language gesture for letter “Y” (Figure 3.4). A proximity forest containing known-class samples is first constructed with 1,000 features per class from 30 classes. Features extracted from 50 individual gesture “Y” images are sequentially added to the proximity forest as references. The testing classification accuracy on 321 samples is determined after adding each sample to the forest. The experiment is repeated 10 times, and each time the 50 reference features are added to the forest in a different random order.

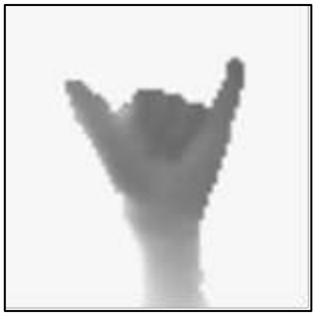

Figure 3.4. A representative depth hand gesture performing the letter “Y” in American Sign Language.

Figure 3.5 shows the mean accuracy with the number of new-class features added to the forest. When there is no reference feature in the forest, the accuracy is 0. However, the accuracy increases rapidly with the number of added new-class features. With 10 reference features in the forest, which contains 30,000 other features from 30 known classes, the classification accuracy is approaching 100%, which is even higher than what the HandNet can achieve in classifying known classes (Table 3.3). This result indicates the ANN algorithm is robust in one-shot learning for classifying hand gestures. In addition, this further indicates that CNN models can generate meaningful feature representations for at least some new-class hand gesture images.

3.2.5 Experiment 5: One-shot learning of images in ImageNet

To test the one-shot learning capabilities of a larger CNN model at a larger scale, the ILSVRC2012 dataset and the Inception v4 model are used. As mentioned in Chapter 2, previously published literature

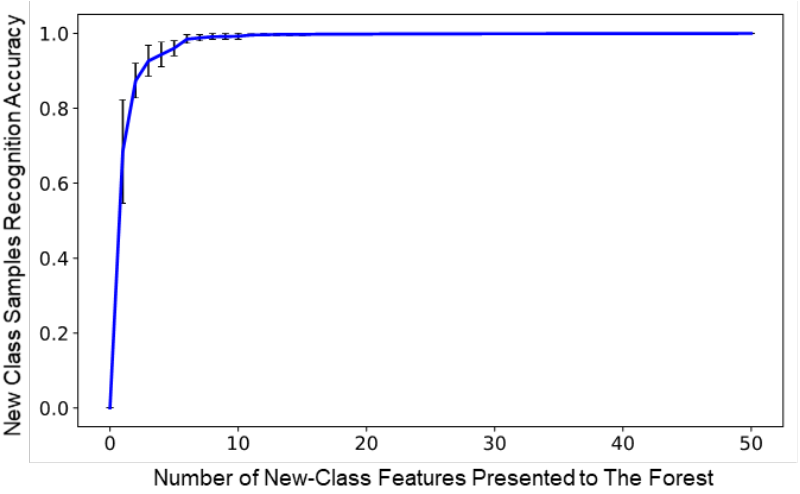

Figure 3.5. One-shot learning of hand gestures showing the mean classification accuracies across 10 experiments (error bars: standard deviation).

tested one-shot learning on ILSVRC2012 dataset by randomly selecting 100 classes and retraining a novel model, so there is no consensus on which dataset to use for large scale one-shot learning with ImageNet. This thesis does not intend to retrain the model and tries to test our algorithm on ILSVRC2012 dataset with full scale, so all classes from the ILSVRC2012 dataset are used and a new ImageNet class is picked for testing. Figure 3.6 shows 3 representative images of the used helmet flower class (Wordnet ID: n11723227, resized according to their original size and scale to fit in page). Similar to other training classes in the ILSVRC2012 dataset, these images have different sizes, shapes, background and scale. There are two other classes of flowers in the ILSVRC2012 dataset: daisy (class index 985) and yellow lady's slipper (class index 986), but they look very different from the helmet flower in terms of shape, color and structure. Therefore, the existence of two flower classes is not expected to highly benefit the one-shot learning of helmet flowers, although it is possible.

A proximity forest containing known-class features is first constructed with randomly chosen 200 features per class from 1,000 classes due to limitation of computation power. Therefore, there are 200,000 known-class features in the constructed proximity forest. Fifty helmet flower images are randomly chosen, and their features are extracted as reference features, which are sequentially added to the proximity forest as references. The classification accuracy on 671 testing samples is determined after

Figure 3.6. Representative helmet flower images from ImageNet.

adding each reference sample. The experiment is repeated 3 times, and each time the reference features are added to the forest in random order.

Figure 3.7 shows the relationship between the mean classification accuracy and the number of new class instances in the forest. With 10 reference features in the proximity forest, the accuracy is approaching 40%, and 50 reference features lead to a classification accuracy of 75%, which is only slightly lower than the accuracy of classifying known classes (Table 3.4). Considering there are 200,000 known-class features in the forest and the helmet flower images have very different scales and sizes, our ANN algorithm has excellent performance. In addition, this indicates that ANN algorithm can perform well in a large-scale dataset for one-shot learning, and CNN models may generate meaningful feature representations for at least some new-class ImageNet images.

3.3 Conclusion

This chapter focuses on the one-shot learning capabilities of CNN models. The negative of Pearson's correlation is chosen as the distance measure for the ANN algorithm because of the smaller

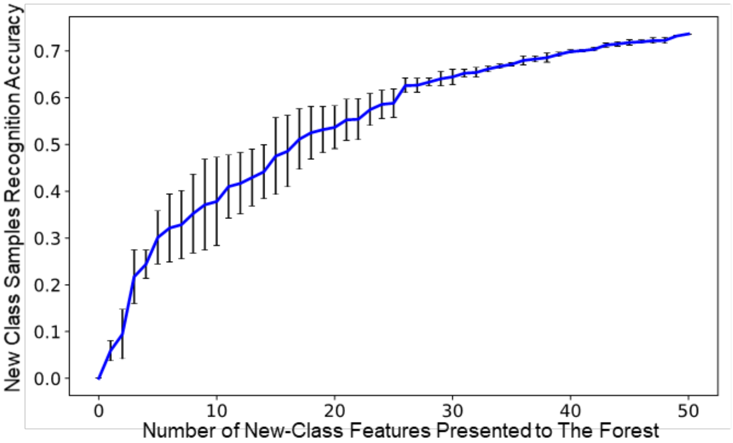

Figure 3.7. One-shot learning of helmet flower images.

intra-class feature distances and the larger inter-class feature distances. The ANN algorithm with CNN features is first shown to be effective in replacing softmax function for classification of known classes. The ANN algorithm has better performance than softmax function for the hand gesture dataset and comparable performance for the ImageNet dataset. This result indicates the ANN algorithm is robust in classifying known classes. In addition, it suggests that the features from the same class tend to cluster together, and those from different classes are further apart. Therefore, the feature vectors may be good representations of their original images. The ANN algorithm is next used for one-shot learning. The results show effective one-shot learning capabilities of the ANN algorithm for both the small-scale gesture dataset and the large-scale ImageNet dataset. This provides a new method for efficient and effective one-shot learning and indicates that CNN models can be generalized to produce meaningful features for at least some untrained new classes. In summary, this chapter proposes an effective one-shot learning algorithm, and indicates that the CNN features are good representations of the images for both known and at least some unknown classes.

Chapter 4 Generalized Curvature Analysis with Handnet

Chapter 3 demonstrates that pre-trained convolutional neural network (CNN) models can be used for effective one-shot learning with the help of an approximate nearest neighbor (ANN) algorithm. This suggests that pre-trained CNN models can generate good representations for samples from known classes and at least some new classes, and the CNN features from the same class cluster together. It is hypothesized that the feature space of CNN models curves around trained classes and the new classes which can be effectively classified using one-shot learning, so that the samples from the same class seem to be clustered together. In addition, the feature space may not curve around samples which do not share the same pattern as trained samples, and these out-of-domain samples cannot be efficiently learned through the one-shot learning algorithm. This chapter focuses on testing these two hypotheses using the generalized curvature analysis (GCA).

Three experiments are used to test the hypotheses. Section 4.2.1 tests the curvature at gesture transitions in a hand gesture video. Section 4.2.2 examines the curvature of out-of-domain samples by using images that are clearly not hand gestures. Finally, Section 4.2.3 attempts to correlate GCA results with one-shot learning capabilities.

4.1 Methods

The HandNet is used as the CNN model because of its smaller scale compared to Inception v4. Section 4.1.1 discusses the data used in this chapter, including a new hand gesture video and out-of-domain samples. Section 4.1.2 describes the details of GCA, and Section 4.1.3 introduces the process for generating a Receiver operating characteristic (ROC) curve to compare histograms.

4.1.1 Data

The hand gesture dataset is the same one used in Chapter 3. To determine the curvature of feature space at transitions of gestures, a hand gesture video of 942 frames performing different gestures is captured using the methods mentioned in Chapter 3.

In addition to test the curvature on gestures images, which are considered as in-domain samples because of its shared patterns with original gesture images, the curvature on out-of-domain samples is also tested. Two types of synthesized out-of-domain input images are used in this chapter: processed ImageNet images and random noise images. Original ImageNet images are colored images with different sizes and scales as mentioned in Chapter 3. These images are first resized so that the length of the shorter side of the original image is 128. The resized images are then center-cropped and converted to grey scale images. Each pixel value of the resulting images is normalized to the range of [-1,1] by first subtracting the center pixel value and then being divided by 255. A random noise picture is synthesized by filling a 128x128 matrix with numbers uniformly randomly sampled from [-1, 1]. Figure 4.1 shows one representative image from each category.

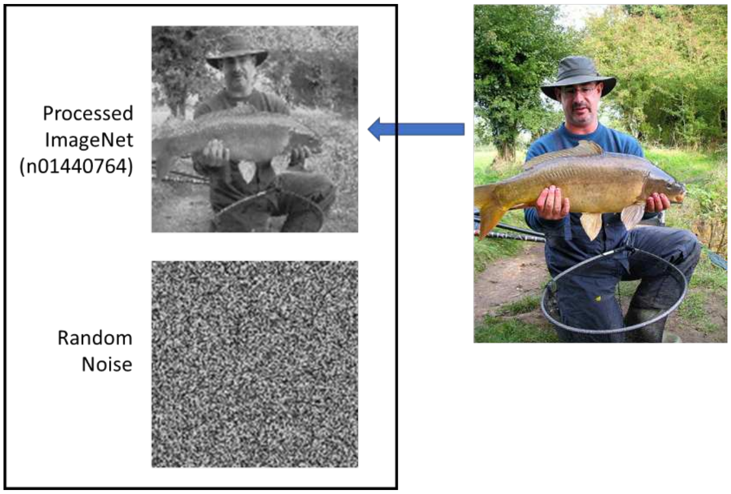

Figure 4.1. Representative processed ImageNet image (and its original picture) and one random noise picture.

4.1.2 Generalized curvature analysis (GCA)

The GCA algorithm is adapted from the original literature (Álvarez-Vizoso *et al.*, 2015; Arn *et al.*, 2018). This thesis only computes the first generalized curvature κ_1. As mentioned in Section 2.4, for a parametric curve γ at a particular point $\gamma(t)$, the curvature function can be expressed as a fixed multiple of a ratio of local singular values at that point. According to equation (2.3), the first generalized curvature for a n-dimensional curve is given by

$$\kappa_1 = \sqrt{\frac{20}{9}\frac{\sigma_2}{\sigma_1^2}} \tag{4.1}$$

where σ_1 and σ_2 are the first and second singular values within the data ball with a size of ϵ, respectively. The ϵ-ball determines the portion of the curve to compute eigenvalues in. For a theoretical continuous curve, the window size ϵ needs to approach 0 so that the ration given by equation (4.1) reflects the actual curvature (Álvarez-Vizoso *et al.*, 2015). However, the feature points in the feature spaces are discrete, making the choice of the data ball size extremely important for determining the right curvature. On one hand, ϵ needs to be small enough so that the equation (4.1) is correct. On the other hand, ϵ needs to be large enough to avoid computing eigenvalues over data noise. To test the curvature for the captured gesture video, this thesis uses a method of dynamically calculating the window size, originally proposed by Arn *et al.* (Arn *et al.*, 2018). Another method of fixing window size is mentioned in Section 4.2.2.

4.1.3 Receiver operating characteristic (ROC) curve

The ROC curve is a graphical representation used to illustrate the binary discriminative capability of a model. The x-axis of a ROC curve is the false positive rate (FPR) and the y-axis is the true positive rate (TPR). The area under curve (AUC) of a ROC curve (AUROC) is used to quantitatively measure the separability, i.e. the capability of distinguishing between binary classes. To generate a ROC curve from several histograms to test their separability, one of the histograms is chosen as a reference, i.e. positive class, and other histograms, considered as negative class, are compared with the reference. The cumulative probabilities of the positive class and the negative class below a certain threshold are considered as TPR and FPR, respectively. By choosing different threshold, one can get multiple pairs of TPR and FPR, thus the ROC curve. This process is mathematically described below.

Given two probability mass function $f_X(t)$ and $f_Y(t)$ for class X and Y, respectively, X is chosen as the positive class. If t_i is chosen as the threshold, the TPR at threshold t_i is given by

$$TPR(t_i) = \sum_{t \le t_i} f_X(t) \tag{4.2}$$

and the FPR is given by

$$FPR(t_i) = \sum_{t \le t_i} f_Y(t) \tag{4.3}$$

The $TPR(t_i)$ and $FPR(t_i)$ are calculated for many different t_i, and their values are plotted (FPR as x-axis and TRP as y-axis) to get the ROC curve.

4.2 Results

4.2.1 Experiment 1: GCA on the features of a temporal sequence of gestures

Chapter 3 suggests that CNN features may be good representations of their original images both trained-class images and at least some new-class images. In addition, the feature space of CNN models may curve around trained classes and the new classes, so that the samples from the same class seem to be clustered together. However, it is not known how the feature space structures around gesture images during class transitions. For example, the features for thumb up seem to cluster together and so are those for thumb down, but it is not clear how does the feature space organize around gesture samples during transitions from thumb up to thumb down. To test this and to better understand the structure of the CNN feature space, GCA is used to analyze the curvature in the feature space.

A new hand gesture video, containing several trained and novel gestures, is first captured. Figure 4.2 shows three representative gesture transitions. In the middle of a transition, the gestures do not look like the ones before or after the transition. Figure 4.3 shows one representative image for each known and new class in the video.

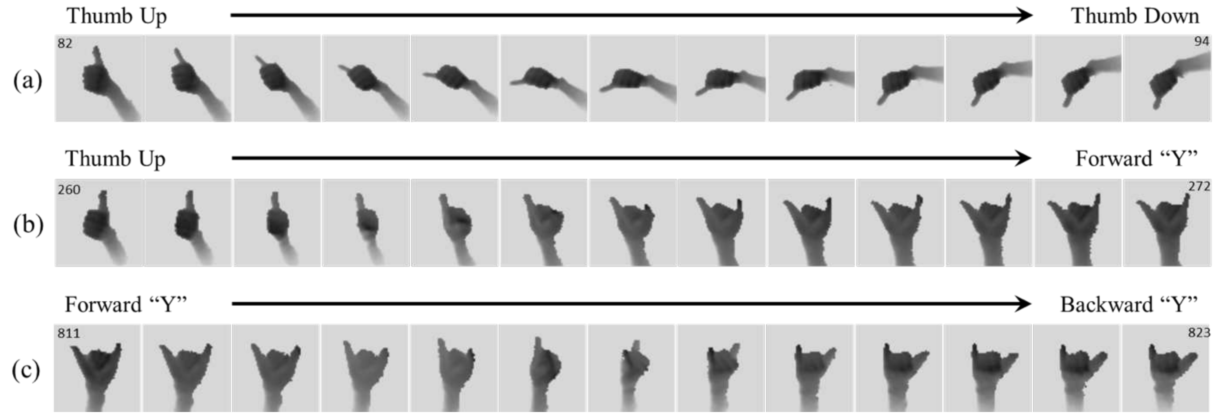

Figure 4.2. Representative gesture transition sequences in a video. These three sequences represent three different scenarios: from a trained class to another trained class (a), from a trained class to a new class (b), and from a new class to another new class (c). Numbers are the frame index in the video.

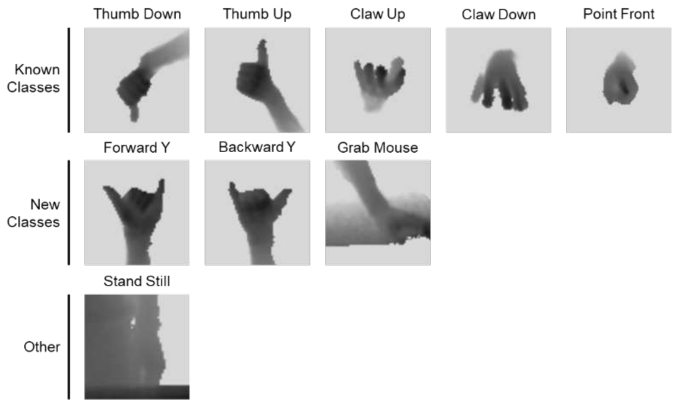

Figure 4.3. Representative images from each class in the video.

Since each frame of the video can be mapped to the feature space as a point, the feature points can be connected based on their temporal sequences to form a curve, and the curvature at each frame/point can be calculated based on their feature representation. The first generalized curvature is calculated as mentioned in methods, and the window size is dynamically determined according to the original reference (Arn *et al.*, 2018).

Figure 4.4 shows the first generalized curvature at each frame. The most striking feature is that the curvature is low at each transition but peaks at some frame between two transitions. This pattern is true irrespective of the number of frames between two transitions. For example, both regions from frame

136 to frame 190 (54 frames) and from frame 230 to frame 264 (34 frames) have this pattern. In addition, both trained classes (blue region) and new classes (forward Y in green region and backward Y in cyan region) share this pattern. Thirdly, all transitions, either from trained class to new class or from a new class to another, have low curvature. Finally, even frames that do not look like regular gestures (see "stand still" frame from Figure 4.3) possess this pattern.

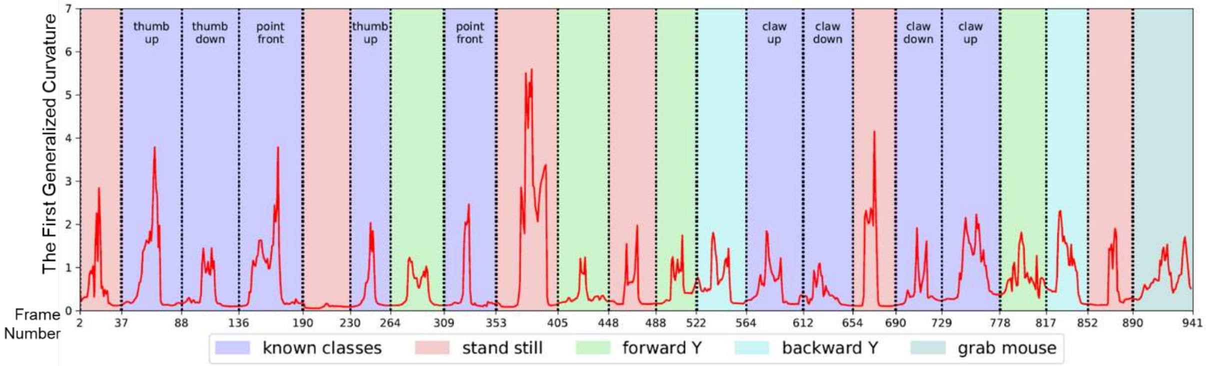

Figure 4.4. The first generalized curvature at each frame of a gesture video. The black dotted line indicates human-labelled transition from one gesture to another. Shaded regions represent frames belonging to the same known/new class.

Two conclusions can be drawn from this experiment. First, curvature is generally high within classes, but low at transitions between classes. This suggests that the feature space curves around each class, and the samples from the same class are densely packed. On the contrary, the transition images seem to be sparsely mapped in between class clusters, and the feature space may not curve around them. Second, the curvature is high within one class irrespective of whether the class is known or a new one. This suggests that the feature space also curves around new classes, although the CNN model has never seen the class during training. In other words, samples from new classes are mapped close to each other in a densely packed region. This provides a theoretical explanation for the success of one-shot learning of new classes. Furthermore, this suggests that CNN models can potentially recognize many new classes, and the feature space curves around these new classes.

4.2.2 Experiment 2: GCA with out-of-domain samples

The previous experiment demonstrates that the CNN feature space curves around both trained and at least some new classes of gestures, which share the same pattern in the sense that they are both depth images captured by Kinect v2. However, it is still not known how the feature space behaves around samples with out-of-domain patterns, and the GCA is used to compute curvature with out-of-domain samples. Because there is no temporal sequence for out-of-domain samples, a certain number of features needs to be sampled to form a data ball so that the curvature can be computed. Based on the result in Appendix B, which shows the dynamically computed window size for each frame corresponding to result in Figure 4.4, the window size, i.e. number of randomly drawn samples, is empirically set to 15 since the window sizes tend to average around 15.

Two classes of synthesized out-of-domain images, processed ImageNet images and random noise images (Figure 4.1), are used. The experiment uses 5 trained/known classes (claw down, claw up, point front, thumb up and thumb down), 2 new classes (forward Y and backward Y), 1 class of processed ImageNet (n01440764) and 1 class of random noise images. For each class, 15 CNN features are randomly sampled and arranged to form a synthesized 15-ball, and the first generalized curvature is computed over the 15-ball at the center sample. This is repeated 1000 times for each class. The results are grouped into known class, unknown class, image net and random, so the four groups have 5000, 2000, 1000 and 1000 data points, respectively. The curvature results for each group are shown as probability density histograms in Figure 4.5.

Based on the histogram, the “known class” and “unknown class” have similar distribution of curvature, and the “image net” and “random” have similar distribution. In addition, the curvature of “known class” and “unknown class” is generally higher than that of the “image net” and “random”. To quantitatively determine the separability between the four groups, a ROC curve is plotted in Figure 4.6. The detailed method for generating a ROC curve from histograms is described in Section 4.1.3. Briefly, the histogram of the group “random” is used as reference or positive class, and the known group, unknown group and ImageNet group are each compared with the reference as a negative class. For each

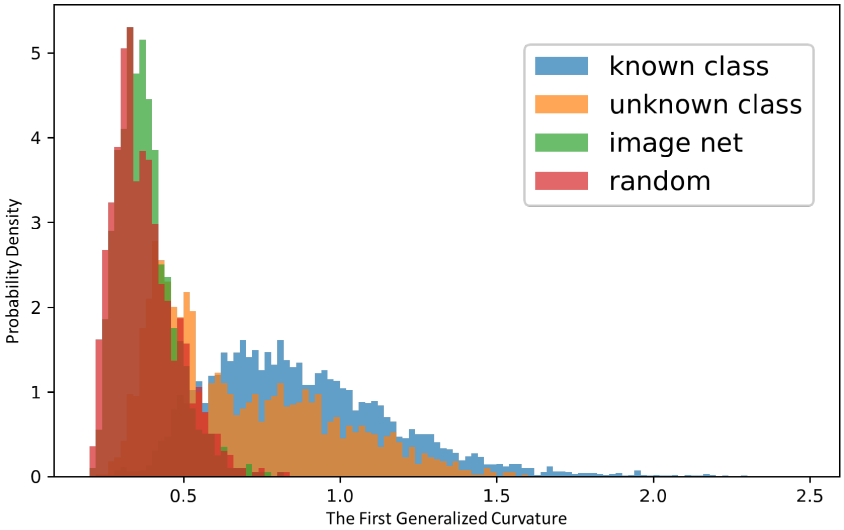

Figure 4.5. Probability density distribution of curvature for each indicated group.

comparison, different curvature values are chosen as threshold, and the cumulative probability less than each threshold is used as the True Positive Rate (TPR, the cumulative probability from random/positive group) or the False Positive Rate (FPR, the cumulative probability from the negative group such as the known group). Different TPR and FPR pairs are plotted to form Figure 4.6.

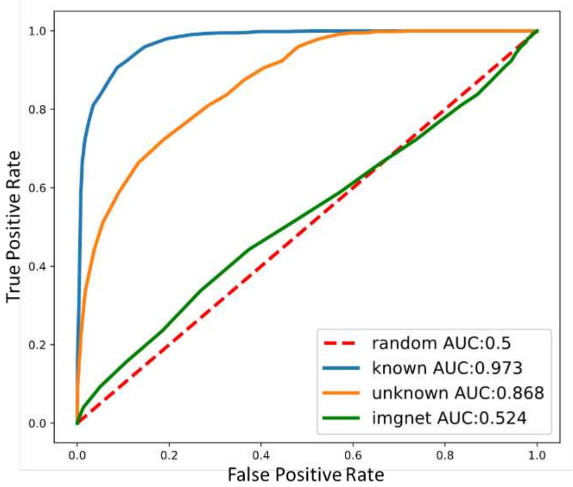

Figure 4.6. ROC curve and AUROC based on Figure 4.5 using probability density distribution of random images as positive reference.

Similar with our intuitive interpretation of the histogram, the image net group and random group have similar distribution. Unknown class and known class have an area under the ROC curve (AUROC) of 0.868 and 0.973, respectively, indicating they have very different distribution from the image net and random group.

These results further indicate that the CNN feature space curves around samples from in-domain classes, consistently with results from Section 4.2.1. In addition, the CNN feature space does not curve around out-of-domain classes. In other words, the CNN features extracted from an out-of-domain class do not cluster together, and the CNN models may not consider those samples belonging to one single class, although humans may do so. From a practical point of view, this experiment also suggests that the curvature analysis could be used to identify out-of-domain images.

4.2.3 Experiment 3: GCA results are correlated with one-shot learning capabilities

If the CNN feature space does not consider an out-of-domain class as a recognizable new one, the features for out-of-domain images will not be good representations, and one-shot learning of an out-of-domain class will not succeed. To test this hypothesis, the CNN models' one-shot learning capability is determined by comparing the results between the in-domain new class group and out-of-domain group. There are 4 classes used in this experiment: backward Y, forward Y, processed n01443537 and processed n01440764. The backward Y and forward Y classes belong to the new class group, and the other two belong to the out-of-domain group. For each class, CNN features from 50 randomly chosen samples are used as reference features that are added into the proximity forest.

The classification accuracy is first determined on new hand gestures. When only the forward Y gesture features are added in the forest, classifying forward Y gesture features achieves 100% accuracy; similarly if only the backward Y gesture features are added in the forest, the accuracy for classifying backward Y class is 99.79% (Table 4.1, column "Only One Class in Forest"). If both forward Y and backward Y features exist in the forest, the classification accuracies for the two classes are still very high with 95.88% and 98.97%, respectively (Table 4.1, column "Both Classes in Forest"). This is consistent with the results of one-shot learning for the HandNet shown in Section 3.2.4, showing that the one-shot learning algorithm has no problem distinguishing between two similar new hand gesture classes.

Table 4.1. New hand gesture classes classification accuracy.

	Only One Class in Forest	Both Classes in Forest
Forward Y Classification Accuracy	100%	95.88%
Backward Y Classification Accuracy	99.79%	98.97%

Next, the classification accuracy is tested on processed ImageNet images. The accuracies when only one class exist in the forest are also high for processed ImageNet images: 96.32% for n01443537 and 95.92% for n01440764 (Table 4.2, column "Only One Class in Forest"). This is not surprising because the features from processed ImageNet images are likely very different from the features from natural gestures. Thus, ANN algorithm can classify one class of processed ImageNet images in a very high accuracy. However, if both n01443537 and n01440764 samples exist in the forest, the classification accuracies are only 63.2% and 54.4%, respectively (Table 4.2, column "Both Classes in Forest"). This indicates the ANN algorithm fails to classify mixed classes of processed ImageNet images, although it can successfully classify mixed classes of natural gestures. In other words, the features of the two processed ImageNet classes are indistinguishable for the ANN algorithm. This is consistent with our hypothesis that the features for out-of-domain images cannot be used as their representations thus are not useful for one-shot learning.

Table 4.2. Processed ImageNet images classification accuracy.

	Only One Class in Forest	Both Classes in Forest
n01443537 Classification Accuracy	96.32%	63.2%
n01440764 Classification Accuracy	95.92%	54.4%

These two results correlate the low curvature with the failed one-shot learning for the out-of-domain samples. This further suggests that the HandNet could not generate meaningful feature representations for out-of-domain samples.

4.3 Conclusion

This chapter uses GCA to probe the structure of CNN feature space, with the goal of better understanding the success of the one-shot learning algorithm and the CNN feature space. It indicates that the CNN feature space curves around in-domain known-class and unknown-class samples, but not samples between class transitions or out-of-domain samples. In addition, it shows that the out-of-domain samples cannot be learned in a one-shot manner, suggesting that a pretrained CNN model could not generate meaningful feature representations for out-of-domain samples. Furthermore, although more investigations are required, the magnitude of the curvature from a randomly sampled 15-ball from one class might be used to identify out-of-domain class and to predict the potentiality of successful one-shot learning of the class. These findings contribute to our understanding of our one-shot learning algorithms and the structure of CNN feature space. Although further analysis and examinations are required, this also provides a potential tool for detecting class transitions, identifying out-of-domain samples and predicting one-shot learning capabilities.

Chapter 5 Conclusion

5.1 Results Discussion

5.1.1 One-shot learning with ANN algorithm

The initial success of using a deep CNN to perform image classification in 2012 has inspired many studies in this area. The exciting results from recent development in CNN models cannot conceal the drawback which is the significant cost of the training process. To make better use of the CNN models, many studies have focused on one-shot learning algorithm with pretrained CNN models to efficiently and effectively classify new classes. This thesis proposes a novel one-shot learning algorithm with adapted ANN algorithm for its simplicity and efficiency.

The first step is to select a good distance measure, and several different metrics were compared. The results showed that the negative of Pearson's correlation is a good distance measure for separating CNN features from different classes since the intra-class pairs of CNN features have smaller correlation-based distances than the inter-class pairs. This work provides foundation for later adaptation of ANN algorithm for one-shot learning.

Before using ANN algorithm with one-shot learning, the ANN algorithm was first tested to determine if it can replace the softmax classifier to classify samples with known classes. The results showed comparable or even better performance of the ANN algorithm than the original softmax function to classify known classes, indicating that the CNN features are good representations for their original images, and the ANN classifier is robust in terms of classifying different known classes.

Finally, the effectiveness of ANN algorithm in one-shot learning was tested in both the hand gesture dataset and ImageNet dataset. The results showed that the ANN algorithm can be effectively used for one-shot learning although the accuracy in ImageNet dataset is slightly lower than that in the hand gesture dataset. The success of using ANN algorithm on one-shot learning further justified that idea that CNN features are good representations of the images. In addition, the CNN models may generate good feature representations not only for images with known classes, but also for those with at least some untrained classes. This indicates that a CNN model is a well-generalized model, probably due to the large amount of training data, and may understand common features based on seen training samples.

The contributions of this part can be summarized as followed:

- Proposed a new effective one-shot learning algorithm.
- Demonstrated that CNN features are good representations for images from both known and at least some new classes.
- Suggested that CNN models are well-generalized and can potentially conceptualize common features from previous training data.

5.1.2 GCA analysis on sets of CNN features

The success of one-shot learning begs the understanding of the reasons behind it. The success of ANN algorithm in classifying known and at least some unknown classes indicates that samples from the same classes are clustered, and those from different classes are further apart. In other words, the CNN feature space curves around trained classes and at least some unseen classes. However, it is not known whether the CNN feature space curves around gestures during class transition or out-of-domain samples which do not share the same pattern as trained images.

The structure of CNN feature space around class transitions was first investigated by determining the curvature of the features from a gesture video at each frame. The transitions of gestures were found to be generally associated with lower curvature, and the samples within the same class were with higher curvature. This is consistent with the idea that CNN feature space curves around known and unseen classes. In addition, it suggests that GCA on CNN features could be used as a tool to identify class transitions in a video.

How feature space structures around out-of-domain samples is tested next. The results showed higher curvature with sets of known-class samples and new-class in-domain samples, and that the curvature with out-of-domain samples are much lower. This suggests that CNN models could not generate meaningful features for samples from out-of-domain classes. In addition, it also suggests that GCA could be used as a tool to detect out-of-domain sample sets.

If a pertained CNN model cannot generate good representations for out-of-domain images, one-shot learning of an out-of-domain class will not succeed. To correlate the results of curvature analysis and the CNN models' one-shot learning capability, the one-shot learning results between the in-domain new class group and out-of-domain group were compared. The results found that out-of-domain samples, which show lower curvature in GCA, cannot be effectively learned in a one-shot manner, but the in-domain samples from unknown classes, which show higher curvature in GCA, can. This further indicates that a pretrained CNN model cannot generate good feature representations for out-of-domain samples. It also suggests that the GCA might be used to predict the ability of a CNN model to one-shot learn certain classes.

The contributions of this part can be summarized as followed:

- Indicated that CNN feature space curves around known and new classes so they seem clustered, but not around samples during transition or out-of-domain samples.
- Indicated that a pretrained CNN model may not generate good feature representations for out-of-domain classes.
- Suggested that GCA could be used to identify class transition in a video.
- Suggested that GCA could be used as a tool to detect out-of-domain samples and to predict the one-shot learning capability of a CNN model on some class.

5.2 Future Directions

Inspired by this work, there are at least two important future questions worth investigating. First, as is shown in the GCA analysis results, the CNN feature space curves around some unseen in-domain classes, and this seems to be correlated with the potential of the CNN model to recognize new classes in one-shot manner despite that no samples from these new classes were used for training. It will be interesting if the feature space can be thoroughly examined to identify the areas which it curves around. Identification of high curvature areas may lead to identification of new classes that can be recognized by CNN models.

Second, the GCA analysis can be used to test if a set of samples have mixed labels, as previous results showed that transitions of hand gestures exhibit low curvature. It is also possible that GCA can be used to detect adversarial samples, which have small perturbations and will be incorrectly classified by CNN models but not by humans (Yuan *et al.*, 2017). Primary results suggested that adversarial samples could be detected using GCA analysis (data not shown), and it is worth a deep investigation as the GCA may be a powerful tool to prevent attacks initiated by adversarial samples.

Bibliography

Álvarez-Vizoso, X. *et al.* (2015) 'Geometry of Curves in $\mathbb R^n$, Singular Value Decomposition, and Hankel Determinants'. Available at: http://arxiv.org/abs/1511.05008 (Accessed: 20 November 2018).

Arn, R. *et al.* (2018) 'Motion Segmentation via Generalized Curvatures', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1. doi: 10.1109/TPAMI.2018.2869741.

Boureau, Y.-L., Ponce, J. and LeCun, Y. (2010) *A Theoretical Analysis of Feature Pooling in Visual Recognition*, *27th International Conference on Machine Learning (ICML-10)*. doi: citeulike-article-id:8496352.

Burgess, J., Lloyd, J. R. and Ghahramani, Z. (2017) 'One-Shot Learning in Discriminative Neural Networks'. Available at: http://arxiv.org/abs/1707.05562 (Accessed: 19 November 2018).

Chellapilla, K. *et al.* (2006) 'High Performance Convolutional Neural Networks for Document Processing To cite this version : High Performance Convolutional Neural Networks for Document Processing'. Suvisoft. Available at: https://hal.inria.fr/inria-00112631 (Accessed: 17 November 2018).

Cui, W. *et al.* (2017) 'CRF-based simultaneous segmentation and classification of high-resolution satellite images', *Global Changes and Natural Disaster Management: Geo-information Technologies*, pp. 33–46. doi: 10.1007/978-3-319-51844-2_3.

Cun, Y. Le *et al.* (2000) 'Handwritten Digit Recognition with a Back-Propagation Network', *Integration The Vlsi Journal*, pp. 20–24. doi: 10.1111/dsu.12130.

Fei-Fei, L., Fergus, R. and Perona, P. (2006) 'One-shot learning of object categories', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4), pp. 594–611. doi: 10.1109/TPAMI.2006.79.

Glorot, X., Bordes, A. and Bengio, Y. (2011) 'Deep Sparse Rectifier Neural Networks', *Jmlr W&Cp*, pp. 315–323. doi: 10.1.1.208.6449.

Gu, J. *et al.* (2015) 'Recent Advances in Convolutional Neural Networks'. Available at: http://arxiv.org/abs/1512.07108 (Accessed: 4 December 2018).

He, K. *et al.* (2015) 'Deep Residual Learning for Image Recognition'. Available at:

http://arxiv.org/abs/1512.03385 (Accessed: 3 December 2018).

Kim, D. *et al.* (2016) 'Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory', *Proceedings - 2016 43rd International Symposium on Computer Architecture, ISCA 2016*, pp. 380–392. doi: 10.1109/ISCA.2016.41.

Konečný, J. and Hagara, M. (2013) 'One-Shot-Learning Gesture Recognition using HOG-HOF Features', *Journal of Machine Learning Research*, 15, pp. 2513–2532. Available at: http://arxiv.org/abs/1312.4190 (Accessed: 19 November 2018).

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012) '1 ImageNet Classification with Deep Convolutional Neural Networks', *Advances In Neural Information Processing Systems*, pp. 1–9. doi: http://dx.doi.org/10.1016/j.protcy.2014.09.007.

Kuehne, H. *et al.* (2011) 'HMDB: A large video database for human motion recognition', in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, pp. 2556–2563. doi: 10.1109/ICCV.2011.6126543.

Lake, B. M. *et al.* (2011) 'Back to Basics: Benchmarking Canonical Evolution Strategies for Playing Atari', *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, pp. 2568–2573. doi: 10.1.1.207.8634.

Lake, B. M., Salakhutdinov, R. and Tenenbaum, J. B. (2013) 'One-shot learning by inverting a compositional causal process', *Advances in Neural Information Processing Systems*, pp. 1–6. doi: 10.1111/j.1471-6402.2010.01567.x.

LeCun, Y. *et al.* (1989) 'Backpropagation Applied to Handwritten Zip Code Recognition', *Neural Computation*, 1(4), pp. 541–551. doi: 10.1162/neco.1989.1.4.541.

Lecun, Y., Bengio, Y. and Hinton, G. (2015) 'Deep learning', *Nature*. Nature Publishing Group, 521(7553), pp. 436–444. doi: 10.1038/nature14539.

Mehrotra, A. and Dukkipati, A. (2017) 'Generative Adversarial Residual Pairwise Networks for One Shot Learning'. doi: 10.1093/mnrasl/slx008.

Nair, V. and Hinton, G. E. (2010) 'Rectified Linear Units Improve Restricted Boltzmann Machines', in

Proceedings of the 27th International Conference on Machine Learning (ICML). USA: Omnipress (ICML'10), pp. 807–14. doi: 10.1.1.165.6419.

Narayana, P. *et al.* (2018) 'Cooperating with Avatars Through Gesture, Language and Action', in. Springer, Cham, pp. 272–293. doi: 10.1007/978-3-030-01054-6_20.

O'Hara, S. and Draper, B. A. (2012) 'Scalable action recognition with a subspace forest', in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1210–1217. doi: 10.1109/CVPR.2012.6247803.

O'Hara, S. and Draper, B. A. (2013) 'Are you using the right approximate nearest neighbor algorithm?', in *Proceedings of IEEE Workshop on Applications of Computer Vision*. IEEE, pp. 9–14. doi: 10.1109/WACV.2013.6474993.

Ojima, Y., Yamaguchi, K. and Taya, M. (2018) 'Quantitative Evaluation of Recombinant Protein Packaged into Outer Membrane Vesicles of Escherichia coli Cells', *Biotechnology Progress*, 34(1), pp. 51–57. doi: 10.1002/btpr.2536.

Papernot, N. and McDaniel, P. (2018) 'Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning'. Available at: http://arxiv.org/abs/1803.04765 (Accessed: 22 December 2018).

Russakovsky, O. *et al.* (2015) 'ImageNet Large Scale Visual Recognition Challenge', *International Journal of Computer Vision*. Springer US, 115(3), pp. 211–252. doi: 10.1007/s11263-015-0816-y.

Sachin Ravi, H. L. (2017) 'Optimization as a model for few-shot learning', in *Iclr*. Available at: https://openreview.net/pdf?id=rJY0-Kcll.

Santoro, A. *et al.* (2016) 'One-shot Learning with Memory-Augmented Neural Networks'. Available at: http://arxiv.org/abs/1605.06065 (Accessed: 21 December 2018).

Simonyan, K. and Zisserman, A. (2014) 'Very Deep Convolutional Networks for Large-Scale Image Recognition'. Available at: http://arxiv.org/abs/1409.1556 (Accessed: 3 December 2018).

Snell, J., Swersky, K. and Zemel, R. S. (2017) 'Prototypical Networks for Few-shot Learning', pp. 4077–4087. doi: arXiv:1703.05175v2.

Strassburger, L. (2006) 'Proof Nets and the Identity of Proofs', in *Proceedings Ninth IEEE International*

Conference on Computer Vision. IEEE, pp. 1134–1141 vol.2. doi: 10.1109/ICCV.2003.1238476.

Szegedy, C. *et al.* (2016) 'Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning'. Available at: http://arxiv.org/abs/1602.07261 (Accessed: 3 December 2018).

Vinyals, O. *et al.* (2016) 'Matching Networks for One Shot Learning'. Available at: http://arxiv.org/abs/1606.04080 (Accessed: 4 December 2018).

Wang, I. *et al.* (2017) 'EGGNOG: A Continuous, Multi-modal Data Set of Naturally Occurring Gestures with Ground Truth Labels', in *Proceedings - 12th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2017 - 1st International Workshop on Adaptive Shot Learning for Gesture Understanding and Production, ASL4GUP 2017, Biometrics in the Wild, Bwild 2017, Heteroge*. IEEE, pp. 414–421. doi: 10.1109/FG.2017.145.

Yuan, X. *et al.* (2017) 'Adversarial Examples: Attacks and Defenses for Deep Learning'. doi: arXiv preprint arXiv:1712.07107.

Appendix A

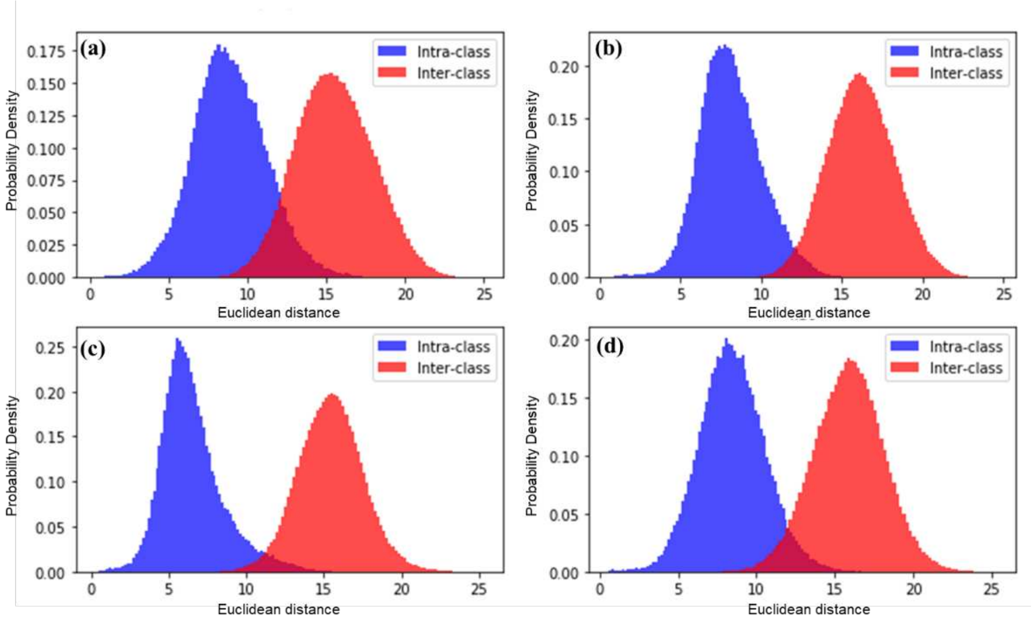

Figure A.1. Histograms showing the inter-class and inter-class sample distances computed using Euclidean distance on CNN features. The hand gesture classes are: **(a)** beckon; **(b)** claw down; **(c)** four front; **(d)** fist.

Appendix B

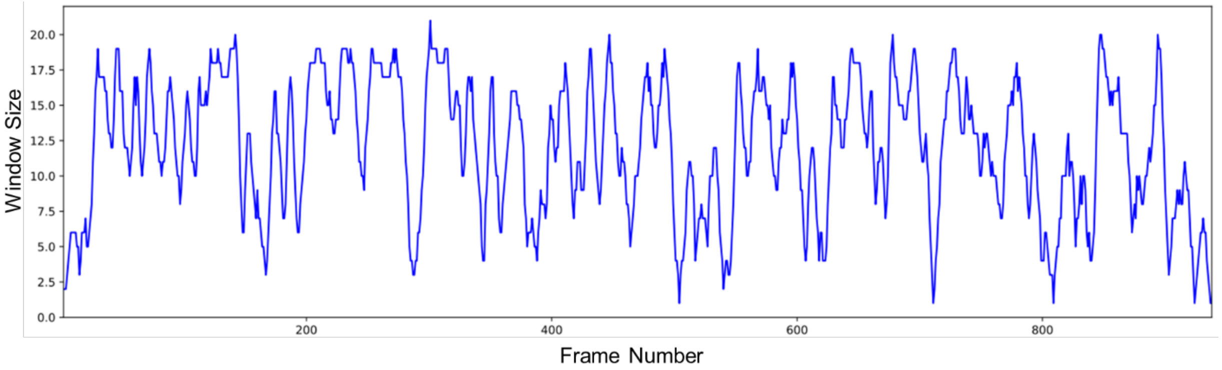

Figure B.1 The dynamically calculated window size at each frame during the calculation of curvature. The curvature result is shown in Figure 4.4.