THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學
Pao Yue-kong Library
包玉剛圖書館

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

IMPORTANT

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

Pao Yue-kong Library, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

http://www.lib.polyu.edu.hk

LEARNING DISCRIMINATIVE MODELS AND REPRESENTATIONS FOR VISUAL RECOGNITION

CAI SIJIA

PhD

The Hong Kong Polytechnic University

2019

The Hong Kong Polytechnic University
Department of Computing

Learning Discriminative Models and Representations for Visual Recognition

Cai Sijia

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

June 2018

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

CAI SIJIA (Name of student)

Abstract

In the past decade, visual recognition systems have witnessed major advances that led to record performances on challenging datasets. However, designing effective recognition algorithms that exhibit robustness to the sizeable extrinsic variability of visual data, particularly when the available training data are insufficient to learn accurate models, is a significant challenge. In this thesis, we focus on designing effective models and representations for visual recognition, via exploiting the characteristics of visual data and vision problems and taking advantages of classic sparse models and state-of-the-art deep neural networks.

The first part of this thesis is dedicated to providing a probabilistic interpretation for general sparse/collaborative representation based classification. With a series of probabilistic modelling for sample-to-sample and sample-to-subspace, we present a probabilistic collaborative representation based classifier (ProCRC) that not only reveals the inner relationship between the coding and classification stages in original framework, but also achieves superior performance on a variety of challenging visual datasets when coupled with the convolutional neural network (CNN) features.

We then facilitate the inherent difficulties in detecting parts and estimating appearance for fine-grained visual categorization (FGVC) problem, we consider the semantic properties of CNN activations and propose an end-to-end architecture based on kernel learning scheme to capture the higher-order statistics of convolutional activations for modelling part interaction. The proposed approach yields more discriminative representation and achieves competitive results on the widely used FGVC datasets even without part annotation.

We also consider weakly-supervised learning of web videos to alleviate the data scarcity issue for video summarization. This is motivated by the fact that the publicly available datasets for video summarization remain limited in size and diversity, making most supervised approaches difficult in learning reliable summarization models. We investigate a generative summarization model via extending the variational autoencoder framework to accept both

the benchmark videos and a large number of web videos. A variational encoder-summarizer-decoder (VESD) is proposed to identify the important segments of raw video using attention mechanism and semantic matching with web video. In this way, our VESD provides a practical solution for real-world video summarization.

We further incorporate sparse models into deep architectures as structured modelling in learning powerful representations from datasets of limited size. The proposed DCSR-Net transforms a discriminative centralized sparse representation (DCSR) model into a learnable feed-forward network which can automatically impose the discriminative structure in data representations. Experiments indicate that DCSR-Net can be regarded as a general and effective module in learning structured representations.

Keywords: Image classification, Fine-grained visual categorization, Video summarization, Supervised learning, Sparse models, Deep neural networks.

List of Publications

Conference Papers

1. **Sijia Cai**, Wangmeng Zuo, Larry S. Davis, Lei Zhang, "Weakly-supervised Video Summarization using Variational Encoder-Decoder and Web Prior". European Conference on Computer Vision (ECCV), Munich, Germany, 2018.

2. **Sijia Cai**, Wangmeng Zuo, Lei Zhang, "Higher-order Integration of Hierarchical Convolutional Activations for Fine-grained Visual Categorization". International Conference on Computer Vision (ICCV), 511–520, Venice, Italy, 2017.

3. **Sijia Cai**, Lei Zhang, Wangmeng Zuo, Xiangchu Feng, "A Probabilistic Collaborative Representation based Approach for Pattern Classication". IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2950–2959, Las Vegas, Nevada, USA, 2016.

Paper Submitted

1. **Sijia Cai**, Wangmeng Zuo, Lei Zhang, Xiangchu Feng, Jianqiang Huang, "Learning A Structured Network for Discriminative Centralized Sparse Representations". Submitted to CVPR 2019.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Lei Zhang, for his continuous guidance, support, encouragement and patience throughout the entire duration of my research. He taught me how best to identify a research problem and how to recognize viable solutions hidden in the thorough analysis of the problem. His critical way of thinking will influence me throughout my industry journey in the coming years.

Besides my supervisor, I would also like to express my appreciation to Prof. Wangmeng Zuo and Prof. Xiangchu Feng for all the constructive advice and endless stream of brilliant ideas throughout this research. Additionally, my sincere thanks also go to Prof. Larry S. Davis, who provided me with an opportunity to join his team, and who gave extensive valuable comments on project related issues.

I thank my supportive lab members at The Hong Kong Polytechnic University. I have to express my special thanks to my office mates: Ms. Jin Xiao, Ms. Yuanyuan Cao, Mr. Zhitao Wang and Mr. Ruohan Zhao. They are also my friends and make my Ph.D. study in Hong Kong colourful and memorable.

Last but not least, I dedicate this small achievement to my family for their love, understanding and patience.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

As a longstanding, fundamental and challenging problem in the fields of computer vision, visual recognition has been a topic of intensive research due to its significance both in understanding the contents of visual data as well as in the critical role it plays in a wide variety of applications. The last decade brought visual recognition to an advanced state, and people realized that feature representation is at the heart of many visual recognition systems. Nevertheless, representation learning is challenging, especially coping with limited-sized visual data, captured under controlled scale, viewpoint, illumination and intra-class variations.

The driving force for feature representation learning in visual recognition research is image classification, which is the main topic in this thesis. Traditionally, numerous efforts have been made to manually propose image features that should be invariant to some degree of variability in appearance changes, and be discriminative against other objects and background in the scene. The widely adopted Bags-of-Words (BOW) feature [147] obtains a histogram of local image descriptors as the image representation. Better feature representations have been achieved by introducing new feature encoding (*e.g.*, Vectors of Locally Aggregated Descriptors (VLAD) [69], Fisher Vectors (FV) [123, 136]) and pooling techniques (*e.g.*, spatial

pyramid [88]). Furthermore, the sparse representation, which introduces the concept of sparsity and represents data as a linear combination of a few elements from a basis or dictionary, has gained great interest in various domains such as face recognition [177], scene categorization [185] and object detection [2]. The sparse representation is expected to have high fidelity to the observed visual content and reveals its underlying structure and semantic information. Due to the powerful sparsity prior, the sparse representation is less likely to become overfitted and therefore requiring much fewer samples for training. From the viewpoint of classification, the sparse representation tends to be formed from samples from its belonging class, which triggers numerous sparse models used in various classification tasks.

With the advent of the big data era, most of the previous engineered features and shallow representation learning techniques become outdated for they cannot learn from the abundant amount of readily-available training samples. Therefore, recent years have seen an explosion of interests on developing models that can extract useful representations from large dataset [1, 135]. Inspired by the visual recognition process in the human cortex, these representation learning models generally follow a multi-layer architecture, also known as deep neural networks (DNNs). A DNN for representation learning consists of a stack of local or non-local feature detectors where simple features (*e.g.*, edges) are detected at lower layers and fed into higher layers for extracting more complex representations (*e.g.*, object parts). The exceptional performance of DNNs can be mainly attributed to their flexibility in representing a rich set of highly non-linear functions [32, 113], as well as the devised methods [63, 67, 91, 116] for efficient training of these powerful networks. Furthermore, employing various regularization techniques [7, 27, 154] ensured that deep models with vast numbers of parameters are statistically desirable in the sense that they will generalize well to unseen data and different tasks. However, DNNs are prone to overfitting problems when trained on relatively small data settings, which limits their potential for representation learning in many recognition applications.

This thesis aims at utilizing sparse models, DNNs as well as modern machine learning

techniques for visual recognition through leveraging the characteristics of visual data and vision problems. This includes a set of new models and algorithms which enable us to incorporate structured priors and domain knowledge into representation learning, therefore, enhance the performance of different visual recognition tasks.

1.2 Visual Representations and Recognition Models

1.2.1 Sparse Models for Visual Recognition

Sparse regularization based representation. The main idea of sparse representation derives from the assumption that a query sample can be represented as a linear combination of an over-complete dictionary where only a few of the dictionary atoms are used in representation. One typical example is the sparse representation based classification scheme which imposes a ℓ_1-norm constraint on the representation coefficients and reported promising results in robust face recognition [177]. Zhang *et al.* [198] further highlighted the importance of collaborative representation and proposed to use ℓ_2-norm to regularize the representation coefficients which achieves similar accuracy but with significantly less computational cost. Many other works proposed different sparsity-related regularizations to improve the quality of the sparse representation while maximally preserving the signal fidelity. For example, Yang *et al.* [185] proposed to use robust sparse coding along with max pooling for image classification and achieved good performance over traditional k-means clustering based method. Liu *et al.* [97] added nonnegative constraint to the sparse representation coefficients. Wang *et al.* [170] used locality constraints during the sparse coding process to speed up computation and coding efficiency. To maintain similarity, Gao *et al.* [40] introduced a Laplacian term for the sparse representation coefficients, which was extended to an efficient algorithm in Cai *et al.* [14]. Besides, Ramirez *et al.* [130] proposed a framework of universal sparse modelling to design sparsity regularization terms. The Bayesian methods were also used for designing the sparsity

regularization terms [70].

Discriminative dictionary based representation. Indeed, discriminative dictionary learning (DDL) has been intensively studied to promote the discriminative power of sparse representation and address the computational drawback for naive sample-based methods. One type of the DDL methods dedicates to improving the discriminative capability of signal reconstruction residual. Rather than learning a dictionary for all classes, these methods exploit structural assumption on dictionary design and impose the learned dictionary with the category-specific property, *e.g.*, learning a sub-dictionary for each class [41, 131, 187]. However, these dictionary learning methods might not be scalable to the problems with a large number of classes. Another type of DDL methods aims to seek the optimal dictionary to improve the discriminative capability of learned representations. These methods learn a dictionary and a classifier concurrently by incorporating some prediction loss on the representation coefficients. In this spirit, Zhang *et al.* [200] extended the original K-SVD algorithm [3] by simultaneously learning a linear classifier. Jiang *et al.* [73, 74] introduced a label consistent regularization to enforce the discrimination of representation coefficients. Mairal *et al.* [102] proposed a supervised dictionary learning scheme by exploiting logistic loss function and further presented a general task-driven dictionary learning framework [101]. Wang *et al.* [175] formulated the dictionary learning problem from a max-margin perspective and learned the dictionary by using a multi-class hinge loss function. Yang *et al.* [187] proposed to adopt both the category-specific strategy for the dictionary and the Fisher discrimination criterion for representation coefficients to enhance class discrimination.

1.2.2 Deep Neural Networks for Visual Recognition

Pre-trained architecture based representation. In recent years, visual representations learned with deep architectures have outperformed hand-crafted or shallow ones in a variety of visual recognition tasks. By carefully designing the network architectures, different deep models

have emerged and lifted the performance evolutionarily. For example, in order to extract visual representations from images, popular convolution neural networks (CNNs) for image classification include AlexNet [84], OverFeat [140], VGGNet [146], GoogLeNet [158, 159], ResNet [61] and DenseNet [66]. With the key differences that the CNN representations are learned directly from data rather than hand-crafted, thus CNNs have a hierarchical architecture learning increasingly abstract levels of representation. Likewise, visual representations from the video have also been intensively studied in deep models recently. Distinct to the image representations, video representations concern not only the spatial information in each video frame, but also the temporal information underlying the frame sequence. To explore the spatial-temporal information from the video data, CNNs based architectures such as 3D CNNs [71] and C3D [163] use 3D convolution and pooling operations to form the spatial-temporal deep representations. Two-stream CNN [145], which consists of spatial network and temporal network for modelling appearance and temporal information respectively, is another popular deep learning framework to extract high-level features from the video due to its good performance and is easy to train [35, 145]. Other generic network architectures for capturing salient features on the temporal coherence representation are Recurrent Neural Networks (RNNs) and the modified Long Short-Term Memory (LSTM) units for long-term dependent and complex video data. One limitation of deep architectures is the demand of a large amount of data for training. Fortunately, studies have shown that the deep representations have good transferability. That is to say, the models trained on a large dataset are ready for use in other related tasks where the data is not enough for training a deep model from scratch. Given a pre-trained deep model, the activations in layers during inference can be treated as high-level representations since they can capture meaningful semantic concepts from visual data.

Fine-tuned architecture based representation. Inspired by the fact that fine-tuning the parameters of a deep model on the small-scale dataset for a specific task can generally improve the performance, researchers have employed this technique to fine-tune the pre-trained deep

models so that they can fit better to the specific data and generate better visual representations for the tasks at hand. The most straightforward to fine-tune a CNN is to modify the last fully connected layer corresponding to the number of classes in the target dataset and generate global representations in an end-to-end manner. In contrast, many works employ fine-tuning on the convolutional layers of a CNN since the global fully connected representations mostly focus on the salient content of visual data but ignore the variation information on clutter and local. One representative kind of methods brings the traditional feature encoding techniques such as VLAD and FV into CNNs [5, 25, 122, 132, 182]. Ruobing *et al.* [178] present a novel pipeline built upon deep CNN features for harvesting discriminative visual objects and parts for scene classification. Dmitry *et al.* [85] propose a DNN topology that incorporates a simple to implement transformation-invariant pooling operator. Gatys *et al.* [43] show that the Gram matrix representations extracted from various layers of VGGNet [146] can be inverted for texture synthesis. Notably, the Gram matrix representation used in their approach is identical to the bilinear pooling of CNN features of Lin *et al.* [94], which is proved to be very effective for fine-grained recognition.

1.3 Key Challenges

Despite advancements in the last decade, both sparse models and DNNs have their inherent limitations in obtaining powerful representations, thus are not capable of performing at a level sufficient to meet the requirements of many computer vision applications. The main difficulty lies in the following aspects:

Interpretability: Informally, interpretability refers to the ability to understand and reason about the model output. However, in spite of continuous research recently, progress in this area remains limited. For example, DNNs have exhibited superior performance in various tasks but continued to be treated mostly as black boxes which provide little human understandable justifications for their outputs and a large number of parameters. Sparse models exploit

the simplicity of the underlying data and produce representations that are readily amenable to human interpretation, while the insights about the inner workings still lack in supervised tasks. We believe that high model interpretability may help people break several bottlenecks for visual recognition, *e.g.*, learning from a few annotations, learning via low-level model parameters, and semantically understanding the representations.

Efficiency: In many real-world applications, recognizing efficiently is as critical as recognizing accurately. Over the last few years, notable progress has been made to boost the accuracy levels of visual recognition, but existing solutions often rely on computationally expensive feature representation and learning approaches. In addition to the opportunities they offer, the extensive visual datasets also lead to the challenge of scaling up while retaining the efficiency of learning approaches and representations. Furthermore, with the prevalence of social media networks and mobile/wearable devices which have limited computational capabilities and storage space, there is a growing need for developing models and visual representations that are fast to compute, memory efficient, and yet exhibiting good discriminability and robustness for visual recognition.

Variability: As a large amount of variability might affect the accurate recognition in real-world tasks, it is very challenging to learn representations of high robustness and distinctiveness. For example, due to the large inter-class variance in conjunction with low intra-class variance in fine-grained visual categorization (FGVC) task, many species in FGVC can only be separated by subtle details, *e.g.*, black vs white colour on the top of a bird's head, which is easy to miss. Fig. 1.1 shows two sets of bird images, where the species of each set are different, although the appearances are similar. Moreover, four images belonging to the same species can have different lighting, pose, viewpoint, deformation, *etc.*, making them look very different.

Data insufficiency: Thanks to the availability of sufficient amount of annotated visual data in datasets such as ImageNet [135] and YouTube-8M [1], existing deep learning representations

Figure 1.1 The species of bird images in green dashed rectangle is Ringed beak gull, and the species of bird images in blue dashed rectangle is California gull.

have been shown to yield high accuracy for visual recognition in recent years. However, there are many notable applications where only limited amounts of annotated training data can be available or collecting labelled training data is too expensive. Such applications impose great challenges to many existing deep learning approaches. When data is scarce, one must rely on general knowledge of the task or use auxiliary data sources, to ensure decent recognition performance.

1.4 Contributions

This thesis explores the design of discriminative models and representations that can offer benefits of robustness to versatility and training insufficiency in visual recognition problems. The different models proposed in this thesis are built upon classic sparse models and state-of-the-art DNNs, and unified by the common goal of identifying and leveraging the discriminative structure inherent in visual data and vision problems thereof. The main contribution of this

thesis could be summarized as follows:

Utilizing appropriate mathematical tools. Exploiting sophisticated machine learning algorithms or finding mathematical properties have found useful in improving the performance of many visual recognition tasks. We carry forward the idea to the case of representation learning. In particular, we establish a probabilistic subspace modelling for both the sparse representation based classification (SRC) and collaborative representation based classification (CRC) schemes, which is very helpful to understand and design shallow representation models. We also design a deep structured network architecture to approximate the iterative solver of the discriminative sparse model to discover the underlying class-oriented structure of vision problem. Furthermore, we consider the FGVC problem from the perspective of kernel learning and revisit the video summarization problem from a generative standpoint, which not only leads to performance improvements but also provide some degree of model interpretability.

Expanding the understanding of vision problems. It is often natural and meaningful to design representation learning architectures according to the characteristics of the vision problem and domain knowledge. For instance, motivated by the higher-order co-occurrence statistics in BOW pipeline, we bring to light this idea and propose a solution for discovering rich part interactions within CNN architectures for FGVC. Notice that the topic-related videos provide visual context to identify the critical parts of the video being summarized, we develop a weakly-supervised approach by exploiting information from extensive collections of web videos to address the insufficient training issue for video summarization task.

Finding properly structured representations. Considering the lack of enough labelled training data in many visual applications, over-fitting is a severe threat for DNNs with a large number of free parameters. In addition, it is also difficult to extract highly discriminative representations from datasets of limited size, mainly due to the lack of cost-effective structured modelling of networks. Motivated by these concerns, we introduced new methods for embedding structures such as global class-centralized structure and the statistical structure of local

CNN features into DNNs. Our comprehensive empirical analysis demonstrated that these regularized networks offer better discrimination and generalization performance compared to conventional deep architectures in representation learning.

Exploiting efficient learning frameworks. Highly efficient algorithms are necessary to extend recognition system to real-world scenarios. In this research, we present computationally efficient techniques to handle learning and training insufficiency in various visual recognition tasks. The primary goal of this thesis is to provide algorithms that can effectively learn discriminative representations in a way without complex modelling and tremendous annotation cost.

1.5 Organization

More specifically, the structure of this thesis is illustrated in Fig. 1.2 and the contributions for each chapters are presented as follows:

Chapter 2 introduces the basic concepts of sparse models and DNNs, and the machine learning tools which are closely connected to the proposed approaches in this thesis.

Chapter 3 explains how the classification mechanism of SRC and CRC can be formulated as probability classification model. Specifically, the chapter develops a probabilistic collaborative representation framework that defines the probabilities of a query sample with respect to a series of subspaces such as collaborative subspace and class-specific subspace. A probabilistic collaborative representation based classifier (ProCRC) is then proposed based on the perspective of maximum likelihood estimation. Extensive experiments on a variety of challenging visual datasets validate the advantages of ProCRC and demonstrate that the proposed model achieves state-of-the-art performance when applied to CNN features. The work in the Chapter has been published at CVPR 2016.

Chapter 4 presents a feature pooling approach in CNNs for FGVC task. FGVC is extremely challenging as it usually needs to identify the semantic object parts to isolate the subtle differ-

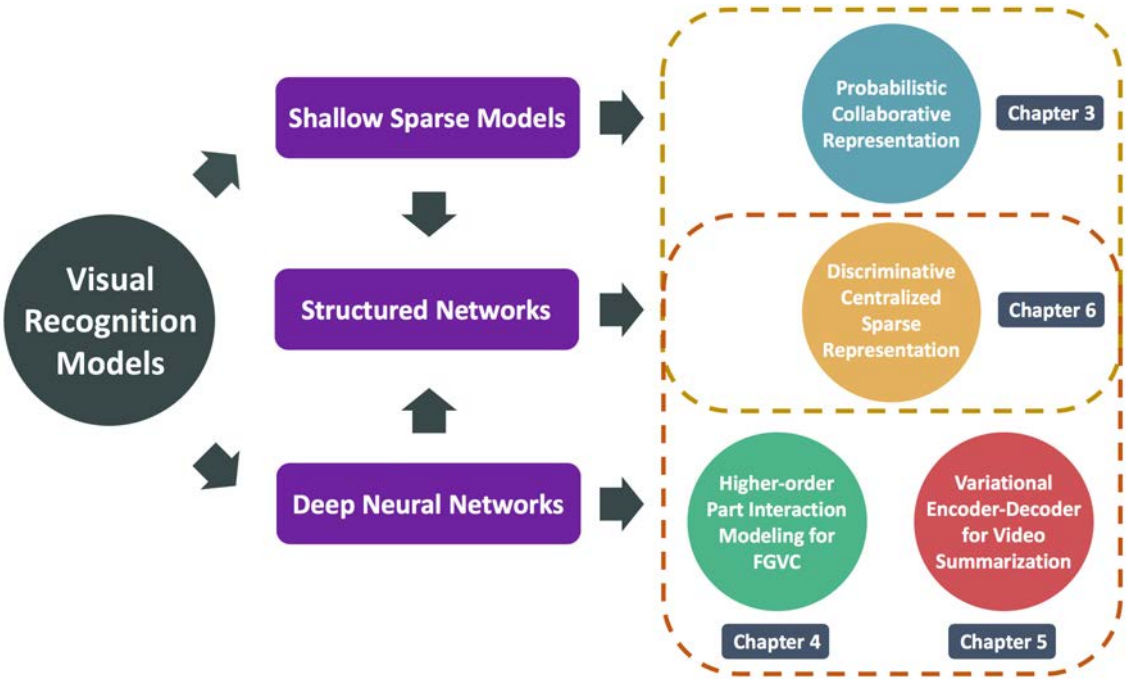


Figure 1.2 The organization of this thesis.

ences among fine-grained categories. This method is based on the observation that deeper convolutional activations can be regarded as the responses of weak semantic parts. In particular, we employ a polynomial predictor to capture higher-order statistics of convolutional activations for modelling rich part interactions from a multi-layer feature fusion scheme in CNNs. In contrast to existing approaches that rely on the modelling of appearance and part annotations, the proposed model only requires image-level labels, yet it still can extract discriminative representations and achieves competitive results on the widely used FGVC datasets. The work in the Chapter has been published at ICCV 2017.

Chapter 5 expands the video summarization problem to leverage user-generated videos from web repositories in a weakly-supervised manner. Recent state-of-the-art deep architectures summarize videos by developing fully supervised approaches on human-crafted temporal importance, which may lead to unreliable models due to the data scarcity issue of current small-size summarization benchmark. Our main idea is to exploit the plentiful web-crawled videos

with only video-level annotations to improve the performance of video summarization. Specifically, we present a generative modelling framework for summarizing videos in the framework of variational encoder-decoder with external web prior. The involved latent video representations maintain the semantic cues from both benchmark data and web data. Furthermore, the overall framework is absorbed into a unified variational encoder- summarizer-decoder (VESD) by introducing a semantic matching loss function with video-level supervision. Experiments are carried out on two challenging and diverse summarization datasets showing that our VESD significantly outperforms existing state-of-the-art methods. The Chapter contains the work that has been accepted by ECCV 2018.

Chapter 6 describes a new computationally efficient framework for learning discriminative representation from datasets of limited size. We combine the merits of both sparse models and neural networks: the structure insights of the optimization-based method and the performance/speed of network-based ones. Specifically, we incorporate the supervised information into a discriminative centralized sparse representation (DCSR) model, and propose a structured network DCSR-Net that implements a truncated form for the iterative scheme of DCSR. DCSR-Net aims to minimize the intra-class variations in the feature space and to learn discriminative representations from limited-sized data. We impose DCSR-Net as a structured regularization into existing CNNs for good discrimination and better supervised fine-tuning. The experiments show that the DCSR-Net helps to improve the performance on classification tasks with training insufficiency issue. The Chapter describes work undergoing review for CVPR 2019.

Chapter 7: Conclusion Finally, we summarize our main contributions and propose the potential future works.

Chapter 2

Preliminaries

In this section, preliminary techniques are explained that will be used in subsequent sections.

2.1 Sparse Models

2.1.1 Sparse and Collaborative Representations for Classification

Representing data as a linear combination of a set of selected known samples has led to promising results in various machine learning applications such as dimensionality reduction. For classification purpose, we have to define the properties and measures to match a query sample against the known, labeled samples. Specifically, denote $\boldsymbol{y}$ the query sample and $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_K]$ the entire training set from K classes, where $\boldsymbol{X}_k$ is the subset of training samples from the k-th class. Some popular richer representations are Sparse Representation (SR) [177] based on solving an ℓ_1-regularized least squares problem, given by:

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}||_2^2 + \lambda||\boldsymbol{\alpha}||_1, \tag{2.1}$$

and Collaborative Representation (CR) [198] based on ℓ_2-regularized least square, solves:

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}||_2^2 + \lambda||\boldsymbol{\alpha}||_2^2, \tag{2.2}$$

where λ is a scalar constant. The classification decision is then performed by assigning the query sample $\boldsymbol{y}$ to the class label with minimum reconstruction error:

$$l(\boldsymbol{y}) = \arg\min_{k} \|\boldsymbol{y} - \boldsymbol{X}_k\hat{\boldsymbol{\alpha}}_k\|_2^2, \tag{2.3}$$

where $\|\boldsymbol{y} - \boldsymbol{X}_k\hat{\boldsymbol{\alpha}}_k\|_2^2$ is the residual error when representing $\boldsymbol{y}$ with samples in the k-th class and $\hat{\boldsymbol{\alpha}}_k$ is the sub-vector of SR/CR $\hat{\boldsymbol{\alpha}}$ associated with the k-th class. SR/CR based classification schema has shown interesting results in face recognition, however, the discriminative information in the training samples is not sufficiently exploited. To address this problem, we can learn properly a dictionary from the original training samples.

2.1.2 Dictionary Learning

The dictionary, which is proposed to represent the encoded sample faithfully, plays an important role in the success of SR and CR. Many discriminative dictionary learning approaches have been proposed to improve the discriminative capability of SR/CR while maintaining the compactness for representation. A general discriminative dictionary learning model can be considered as:

$$< \hat{\boldsymbol{D}}, \hat{\boldsymbol{A}} >= \arg\min_{\boldsymbol{D},\boldsymbol{A}} \mathcal{R}(\boldsymbol{X}, \boldsymbol{D}, \boldsymbol{A}) + \lambda_1\|\boldsymbol{A}\|_p^p + \lambda_2\mathcal{L}(\boldsymbol{A}), \tag{2.4}$$

where $\boldsymbol{X}$ is the training set, $\boldsymbol{D}$ is the dictionary to be learnt, $\boldsymbol{A}$ is the set of representations over $\boldsymbol{D}$, λ_1 and λ_2 are the trade-off parameters, $\mathcal{R}(\boldsymbol{X}, \boldsymbol{D}, \boldsymbol{A})$ is the reconstruction term (*e.g.*, $\|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{A}\|_F^2$), p denotes the parameter of the ℓ_p-norm regularizer (*e.g.*, ℓ_1-norm or ℓ_2-norm), and $\mathcal{L}(\boldsymbol{A})$ denotes the discrimination term for $\boldsymbol{A}$. Usually, each column $\boldsymbol{d}_j$ of the dictionary is required to satisfy $\|\boldsymbol{d}_j\|_2 \leq 1$. Different settings of reconstruction term and discrimination term are proposed in recent years and a common approach to minimize the above objective function is the alternating updating framework, *i.e.*, minimizing one and while keeping the other fixed.

2.2 Deep Neural Networks

2.2.1 Convolutional Neural Networks

The CNN is one of the most notable neural network architectures where multiple layers can be trained in an end-to-end manner. A CNN is ideally suitable for processing static data such as images and can effectively learn complicated mappings from raw inputs to the target, which is able to extract complex representations compared to handcrafted features and shallow learning frameworks.

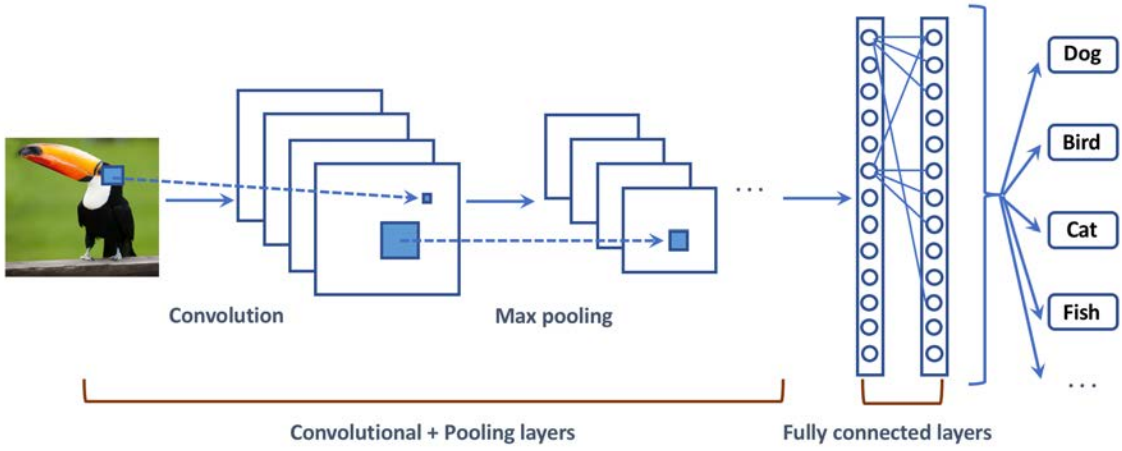


Figure 2.1 The general architecture of a CNN for image classification.

Fig. 2.1 shows the general architecture of a CNN for image classification. A typical CNN structure consists of three main neural layers, which are convolutional layers, pooling layers, and fully connected layers. The convolutional layer applies the convolution operation to filter inputs for useful information; The pooling layer downsamples the inputs using a given selection method; The fully connected layer combines all the outputs of the previous layer and generates the feature representations. We present the detailed operations of these three types of layers as below:

- **Convolutional layer:** Given a three-dimensional input feature map $\boldsymbol{x} \in \mathbb{R}^{H \times W \times D}$, the

convolutional layer convolves $\boldsymbol{x}$ with learned filters $\boldsymbol{f} \in \mathbb{R}^{H' \times W' \times D \times D''}$, and outputs $\boldsymbol{y} \in \mathbb{R}^{H'' \times W'' \times D''}$; *i.e.*,

$$y_{i'',j'',d''} = b_{d''} + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D'} f_{i',j',d,d''} x_{i''+i'-1,j''+j'-1,d}, \tag{2.5}$$

where $b_{d''}$ denotes the bias, and $H'' = 1 + H - H'$ and $W'' = 1 + W - W'$. Generally, a convolutional layer is followed by some activation functions such as the Rectified Linear Units (ReLUs) to impose the nonlinearity.

- **Pooling layer:** The goal of a pooling layer is to provide rotational/position invariance and reduce the dimensions of feature maps and network parameters. Given a feature map $\boldsymbol{x}$, the pooled representation is given by

$$y_{i'',j'',d} = P(\{x_{i''+i'-1,j''+j'-1,d}\}_{1 \le i' \le W', 1 \le j' \le H'}), \tag{2.6}$$

  where W' and H' denote respectively the width and heights of the pooling regions, and P denotes the pooling operator. The most widely-adopted pooling operators are max pooling and average pooling.

- **Fully connected layer:** At the end of alternating convolutional and pooling layers, there are several fully connected layers converting the 2D feature maps into a 1D feature vector, for further feature representation in prediction of the required classification output.

To efficiently train the CNN model, the standard method is using Stochastic Gradient Descent (SGD) with backpropagation algorithm. That is, the prediction output is used to compute the loss with the ground-truth labels. Then based on the loss, the backward step computes the gradients of CNN parameter with chain rules.

Fine-tuning: One advantage of neural networks is that one can refine a pre-trained neural network to new tasks especially when only a small number of training samples are available. The so-called fine-tuning technique first removes the last output layer of a neural network

and attaching a new layer with randomly initialized parameters. Then we can train these new parameters efficiently and achieve good performance on the new task. Because the pre-trained model has already learned comprehensive representations through millions of training samples, the fine-tuning procedure can start by exploring useful representations for the new task without going all the way from scratch.

2.2.2 Recurrent Neural Networks

The RNN is a popular architecture that has been widely used for processing sequential inputs such as video frames. The RNN model has a recurrent temporal loop that can capture compositional representations in the time domain and is suitable for modelling the dynamics of sequential inputs. The recurrent temporal loop creates a deep structure in the RNN model when unfolding in the time series. Fig. 2.2 shows the structure of RNN and the unfolded network.

Given a sequence $\boldsymbol{x} = \{x_1, x_2, \ldots, x_T\}$, an RNN computes a sequence of hidden states $\boldsymbol{h} = \{h_1, h_2, \ldots, h_T\}$ and output $\boldsymbol{y} = \{y_1, y_2, \ldots, y_T\}$ as follows:

$$h_t = \mathcal{H}(W_{ih}x_t + W_{hh}h_{t-1} + b_h) \tag{2.7}$$

$$y_t = \mathcal{O}(W_{ho}h_t + b_o), \tag{2.8}$$

where W_{ih}, W_{hh}, W_{ho} denote weight matrices, b_h, b_o denote the biases, and $\mathcal{H}(\cdot)$ and $\mathcal{O}(\cdot)$ are the activation functions of the hidden layer and the output layer, respectively. Typically, the activation functions are defined as logistic sigmoid functions.

The traditional RNN is hard to train due to the so-called vanishing gradient problem. If the input sequence is too long, the gradient update through backpropagation becomes inefficient. The weight updates decrease exponentially with the number of backpropagation steps, which makes the training extremely slow. This problem limits the maximum length of sequences that an RNN can accept.

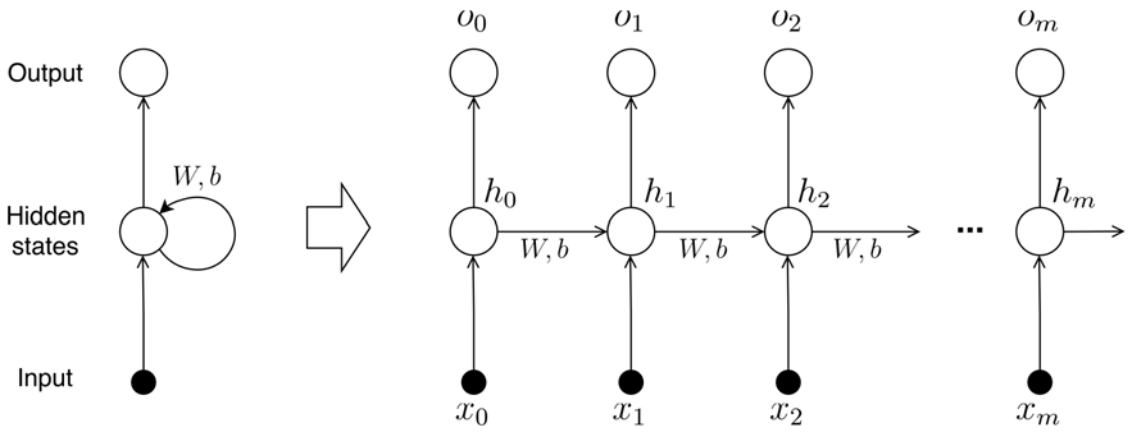


Figure 2.2 An Recurrent Neural Network and the unfolded structure.

To alleviate the vanishing gradient problem, the Long Short-Term Memory (LSTM) model is then proposed by Hochreiter and Schmidhuber [64]. Specifically, in addition to the hidden state h_t, the LSTM also includes an input gate i_t, a forget gate f_t, an output gate o_t, and the memory cell c_t (shown in Fig. 2.3. These gates are regularized with sigmoid functions and control the portion of information passed through the update functions. To be specific, in this architecture i_t and f_t are sigmoidal gating functions, and these two terms learn to control the portions of the current input and the previous memory that the LSTM takes into consideration for overwriting the previous state. Meanwhile, the output gate o_t controls how much of the memory should be transferred to the hidden state. These mechanisms allow LSTM networks to learn temporal dynamics with long time constants.

The hidden layer and the additional gates and cells are updated as follows:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{2.9}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2.10}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.11}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{2.12}$$

$$h_t = o_t \tanh(c_t) \tag{2.13}$$

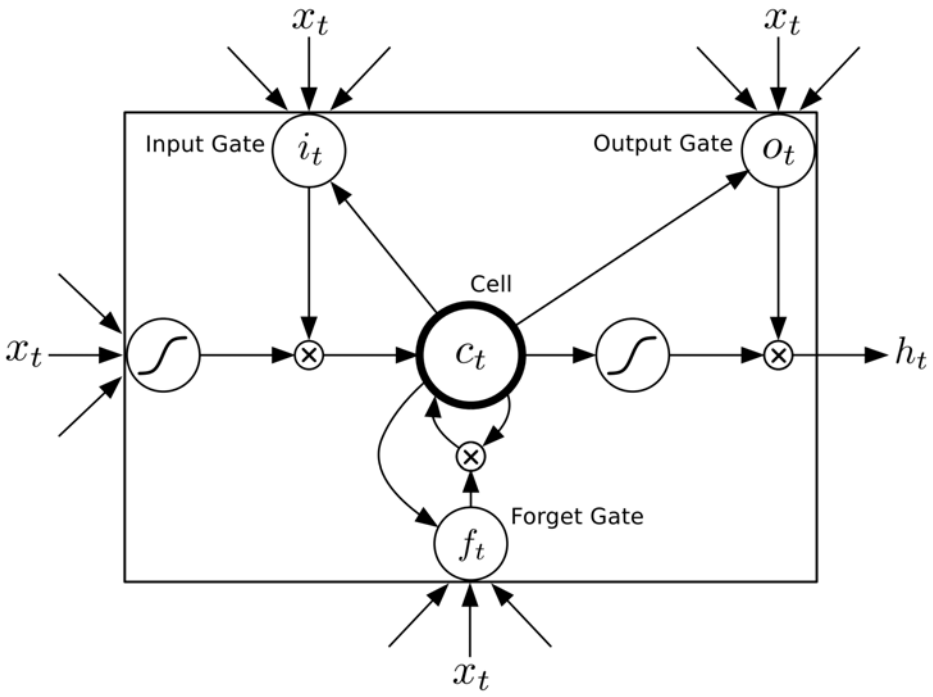


Figure 2.3 A diagram of a LSTM memory cell (adapted from Graves *et al.* [50]).

One shortcoming of conventional RNNs is that they are only able to make use of previous context. To further exploit future context, bidirectional RNNs [139] are then proposed to process the data in both directions with two separate hidden layers, which are then fed forwards to the same output layer. As illustrated in Fig. 2.4, a bidirectional RNN computes the forward hidden sequence $\overrightarrow{\boldsymbol{h}}$, the backward hidden sequence $\overleftarrow{\boldsymbol{h}}$ and the output sequence $\boldsymbol{y}$ by iterating the backward layer from $t = T$ to 1, the forward layer from $t = 1$ to T and then updating the output layer:

$$\overrightarrow{h}_t = \mathcal{H}(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \tag{2.14}$$

$$\overleftarrow{h}_t = \mathcal{H}(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}) \tag{2.15}$$

$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y. \tag{2.16}$$

Combing bidirectional RNNs with LSTM gives bidirectional LSTM [51], which can access long-range context in both input directions.

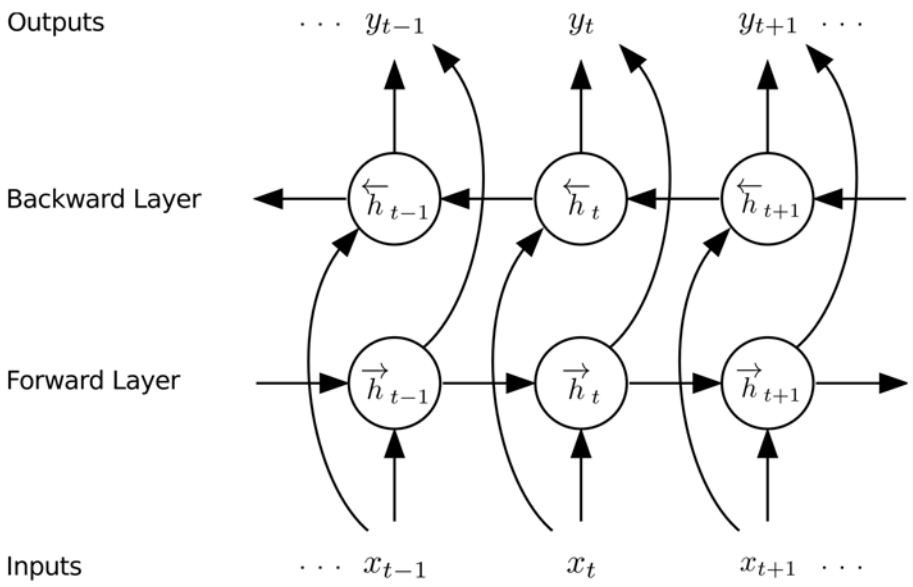


Figure 2.4 A diagram of bidirectional RNN (adapted from Graves *et al.* [50]).

2.2.3 Variational Auto-encoder

Auto-encoders are neural networks used for unsupervised representation learning. The dimension of the input is equal to the dimension of the output, and the purpose of the network is to reconstruct the input. The representation that is learned at the bottleneck of the network, is called the code or latent space. An auto-encoder can be divided in two distinct modules, the encoder and the decoder. The encoder is a function that maps an input $\boldsymbol{x}$ into some latent representation z, $enc : \mathcal{X} \rightarrow \mathcal{Z}$. The decoder maps the latent representation to the input space, $dec : \mathcal{Z} \rightarrow \mathcal{X}$. The objective of the auto-encoder is to minimize some distance loss as defined:

$$dec^*, enc^* = \arg\min_{enc,dec} \|\boldsymbol{x} - dec(enc(\boldsymbol{x}))\|_2^2 \tag{2.17}$$

Variational Auto-encoders (VAEs) [80] assume some generative process from the latent space z to $\boldsymbol{x}$ (depicted in Fig. 2.5. Note that the latent variable z is treated as a random variable.

By introducing a variational distribution $q_\theta(z|\boldsymbol{x})$, a lower bound for $p(\boldsymbol{x})$ can be derived with Jensen's inequality in Eqn. (2.18). In this equation, KL represents the Kullback-Leibler

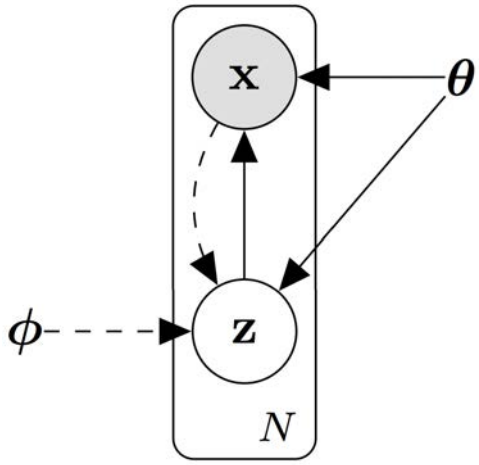


Figure 2.5 The graphical model of generative process for the Variational Auto-encoder.

divergence, probability distributions are parametrized by $\boldsymbol{\theta}$, and the variational distribution is parametrized by $\boldsymbol{\phi}$. The decoder is now defined as the conditional distribution $dec := p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$. The encoder is defined as the variational distribution $enc := q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. A common assumption is to let $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ be a multivariate normal distribution with diagonal variances. Thus, the encoder is defined as $enc := \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}\boldsymbol{I})$. Then the choice of prior is often $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} \\
&= \log \int q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} \\
&\geq \int q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} \\
&= \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] + \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) || p_{\boldsymbol{\theta}}(\boldsymbol{z}))
\end{aligned} \tag{2.18}$$

Chapter 3

A Probabilistic Collaborative Representation based Approach for Visual Classification

3.1 Introduction

Visual classification is one of the fundamental problems in computer vision and machine learning. Given a set of training samples $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_K]$, where $\boldsymbol{X}_k, k = 1, 2, \ldots, K$, is the sample matrix of class k, visual classification aims to predict the class label of a query sample $\boldsymbol{y}$. Many visual classification schemes have been proposed in the past decades. Generally speaking, there are two categories of visual classification methods [10, 114]: parametric methods and non-parametric methods. The parametric visual classification methods (*e.g.*, SVM) focus on how to learn the parameters of a hypothesis classification model from the training data. The learned parametric model is then used to predict the class labels of unknown data. In contrast, the non-parametric visual classification methods (*e.g.*, nearest neighbor) do not learn a parametric model for classification but use the training samples directly to predict the class

labels of unknown data. Though non-parametric methods bear some weaknesses in computational efficiency, recent works have revealed their advantages (*e.g.*, avoid over-fitting) over the parametric based methods [10, 195].

A popular type of non-parametric classifiers which are widely used in various visual recognition tasks is the distance based classifiers, *e.g.*, the nearest subspace classifier (NSC) [22]. The principle of such classifier is to assign a test sample to the class which has the shortest distance to it. However, the distance based non-parametric classifiers rely heavily on the pre-determined distance or similarity metrics. Though some commonly used metrics, such as Euclidean distance, manifold distance and principal angle based correlation [59, 171], are intuitive to describe the variations among samples, they have limitations in accurately reflecting the intrinsic similarity among objects [107]. In order to better characterize the similarity, a promising choice is to introduce the uncertainties of the outputs of a classifier for decision making, as what has been done in probabilistic SVMs [38, 92, 125]. Probabilistic SVM estimates the posterior probabilities of class labels by the calibration techniques, such as Platt's scaling [92, 125] which transforms the classifier's scores into the calibrated probabilities over classes by fitting a sigmoid posterior model.

An alternative approach to probabilistic SVM is the probabilistic subspace methods, *e.g.*, probabilistic principal component analysis (PPCA) [87, 162] and probabilistic linear discriminant analysis (PLDA) [128], which reformulate the subspace methods as a latent variable model and optimize the parameters via maximum likelihood estimation. Therefore, the probabilistic subspace methods can be used to better model the class-conditional densities in classification. Moghaddam and Pentland [109, 110] proposed to utilize a probabilistic similarity measure to model the probability distribution of subspace spanned by the changes of an object's appearance. Wang *et al.* [171] further extended the probabilistic distance measure from two images to two linear subspaces (image sets), and formulated it as a Bayesian face recognition framework [108]. However, most probabilistic subspace methods make strong

assumptions on the distribution of noise and do not provide a straightforward procedure for multi-subspace cases.

How to represent the test sample is a key issue in distance based non-parametric classifiers. In SRC classifier proposed by Wright *et al.* [177], a test sample is approximated by a linear combination of training samples from all classes with ℓ_1-norm sparsity regularization on the representation coefficients. In [198], Zhang *et al.* argued that the success of SRC should be largely attributed to the collaborative representation of a test sample by the training samples across all classes. They further proposed an effective CRC classifier by utilizing ℓ_2-norm regularizer. The SRC/CRC classifiers can be regarded as distance based classifiers since they classify a test sample based on the shortest Euclidean distance from it to each class. Many modifications of SRC/CRC have been proposed for face recognition and other visual recognition tasks [20, 21, 28, 72, 170, 185, 192]. Chi and Porikli [20, 21] suggested a collaborative representation optimized classifier (CROC) to combine NSC and SRC/CRC for multi-class classification. Despite the fact that many variants, improvements and applications of SRC/CRC have been proposed, there still lacks a substantial understanding of the classification mechanism of them. Though an inspiring geometric interpretation of CRC has been given in [198], this interpretation is not informative enough to reveal the intrinsic reason of CRC's success.

Motivated by the work of probabilistic subspace methods [107, 109, 110], in this work we analyze the classification mechanism of CRC from a probabilistic viewpoint and propose a probabilistic collaborative representation based approach for visual classification. First, we present a probabilistic collaborative representation framework, where the probability that a test sample belongs to the collaborative subspace of all classes can be well defined and computed. Very interestingly, this probabilistic collaborative representation framework explains clearly the ℓ_2-norm regularized representation scheme used in CRC. Consequently, we present a probabilistic collaborative representation based classifier, which jointly maximizes the likelihood

that a test sample belongs to each of the multiple classes. The final classification is performed by checking which class has the maximum likelihood. Our extensive experiments on various visual classification tasks demonstrate that ProCRC outperforms many commonly used classifiers, including SVM, kernel SVM, SRC, CRC and CROC.

3.2 Probabilistic Collaborative Subspace Representation

3.2.1 Probabilistic Collaborative Subspace

Suppose that we have a collection of training samples from K classes $\boldsymbol{X} = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_K]$, where $\boldsymbol{X}_k$ is the data matrix of class k and each column of $\boldsymbol{X}_k$ is a sample vector. We view $\boldsymbol{X}$ as the data matrix of an expanded class, and denote by $l_{\boldsymbol{X}}$ the label set of all candidate classes in $\boldsymbol{X}$. Denote by $\mathcal{S}$ the linear subspace collaboratively spanned by all samples in $\boldsymbol{X}$. Then for each data point $\boldsymbol{x}$ in the collaborative subspace $\mathcal{S}$, it can be represented as a linear combination of samples in $\boldsymbol{X}$: $\boldsymbol{x} = \boldsymbol{X}\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is the representation vector.

Since $\boldsymbol{X}$ involves many samples from all classes, the collaborative subspace $\mathcal{S}$ is much bigger than the subspace spanned by each individual class $\boldsymbol{X}_k$. Therefore, though all data points $\boldsymbol{X}\boldsymbol{\alpha}$ fall into $\mathcal{S}$, we argue that their confidences to be labeled as $l_{\boldsymbol{X}}$ should be different, depending on how the representation vector $\boldsymbol{\alpha}$ is composed. Let us use an example to explain the idea. As illustrated in Fig. 3.1, $\boldsymbol{X}$ is a collection of face images from different subjects, and then $l_{\boldsymbol{X}}$ is a label set of face subjects. With vector $\boldsymbol{\alpha}_1 = [0.24,\ 0.22,\ 0.11,\ 0.21,\ 0.13,\ 0.10]$, a face image $\boldsymbol{x}_1 = \boldsymbol{X}\boldsymbol{\alpha}_1$ is composed, and with vector $\boldsymbol{\alpha}_2 = [-0.65,\ 0.46,\ 0.58,\ 0.65,\ -0.42,\ 0.36]$, another face image $\boldsymbol{x}_2 = \boldsymbol{X}\boldsymbol{\alpha}_2$ is composed. Clearly, $\boldsymbol{x}_1$ is more likely to be a face image than $\boldsymbol{x}_2$, and it should have higher confidence to be labeled as $l_{\boldsymbol{X}}$.

From the example in Fig. 3.1, we can see that the representation vector $\boldsymbol{\alpha}$ determines the confidence that $\boldsymbol{x}$ belongs to $l_{\boldsymbol{X}}$. With a more detailed look of vectors $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$, we can see that $\boldsymbol{\alpha}_1$ contains smaller coefficients (in terms of magnitude), which make $\boldsymbol{x}_1$ approach to the

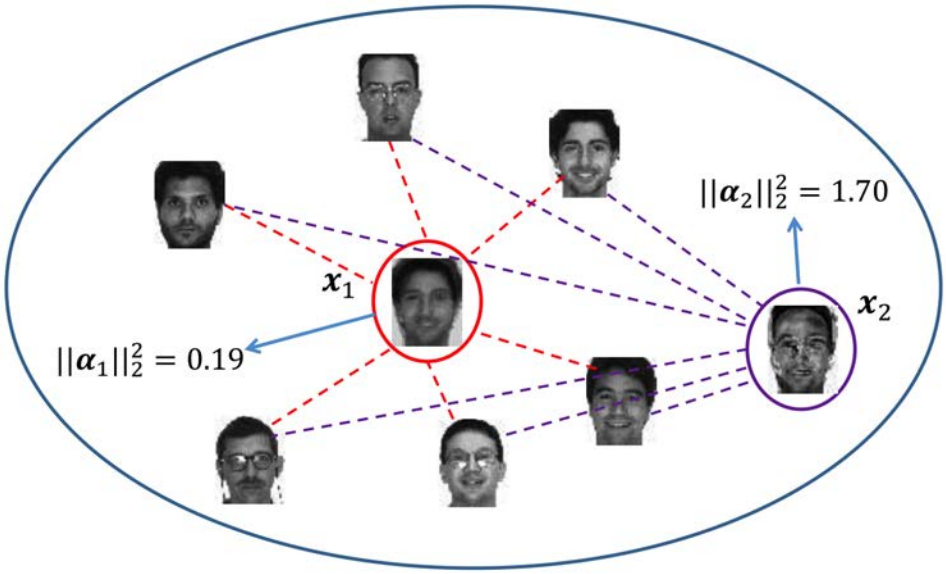


Figure 3.1 Illustration of probabilistic collaborative subspace. $\boldsymbol{x}_1$ has a smaller ℓ_2-norm of its representation vector, and is more likely to be a face image than $\boldsymbol{x}_2$.

center area of subspace $\mathcal{S}$, while $\boldsymbol{\alpha}_2$ has relatively bigger coefficients, making $\boldsymbol{x}_2$ approach to the boundary area of $\mathcal{S}$. Based on these observations, we propose to formulate $\mathcal{S}$ as a probabilistic collaborative subspace; that is, different data points $\boldsymbol{x}$ have different probabilities of $l(\boldsymbol{x}) \in l_X$, where $l(\boldsymbol{x})$ means the label of $\boldsymbol{x}$, and $P(l(\boldsymbol{x}) \in l_X)$ should be higher if the ℓ_2-norm of $\boldsymbol{\alpha}$ is smaller, vice versa. One intuitive choice is to use a Gaussian function to define such a probability:

$$P(l(\boldsymbol{x}) \in l_X) \propto \exp(-c\|\boldsymbol{\alpha}\|_2^2), \tag{3.1}$$

where c is a constant. With Eqn. (3.1), we call the subspace $\mathcal{S}$ a probabilistic collaborative subspace, whose data points are assigned different probabilities based on $\boldsymbol{\alpha}$.

3.2.2 Probabilistic Representation Outside the Collaborative Subspace

Eqn. (3.1) defines the probability of a data point inside the collaborative subspace $\mathcal{S}$. In practice, the test sample $\boldsymbol{y}$ usually lies outside the subspace $\mathcal{S}$. In order to measure the probability

that $\boldsymbol{y}$ belongs to l_X, *i.e.*, $P(l(\boldsymbol{y}) \in l_X)$, we could find a data point $\boldsymbol{x}$ in $\mathcal{S}$, and then compute two probabilities: $P(l(\boldsymbol{x}) \in l_X)$ and the probability that $\boldsymbol{y}$ has the same class label as $\boldsymbol{x}$, *i.e.*, $P(l(\boldsymbol{x}) = l(\boldsymbol{y}))$. With $P(l(\boldsymbol{x}) \in l_X)$ and $P(l(\boldsymbol{x}) = l(\boldsymbol{y}))$, we can readily have:

$$P(l(\boldsymbol{y}) \in l_X) = P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) \in l_X) \cdot P(l(\boldsymbol{x}) \in l_X). \tag{3.2}$$

$P(l(\boldsymbol{x}) \in l_X)$ has been defined in Eqn. (3.1). $P(l(\boldsymbol{x}) = l(\boldsymbol{y}) | l(\boldsymbol{x}) \in l_X)$ can be measured by the similarity between $\boldsymbol{x}$ and $\boldsymbol{y}$. Here we adopt the Gaussian kernel (a.k.a heat/radial basis function kernel) to define it:

$$P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) \in l_X) \propto \exp(-\kappa \|\boldsymbol{y} - \boldsymbol{x}\|_2^2), \tag{3.3}$$

where κ is a constant. Gaussian kernel is a widely used measure to characterize the neighbor-based similarity of two vertices in graph, and its advantages have been observed in many real-world applications such as data reduction [58], face analysis [62] and image clustering [203].

With Eqn. (3.1)~Eqn. (3.3), we have

$$P(l(\boldsymbol{y}) \in l_X) \propto \exp(-(\kappa \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + c\|\boldsymbol{\alpha}\|_2^2)). \tag{3.4}$$

In order to maximize the probability $P(l(\boldsymbol{y}) \in l_X)$, we can apply the logarithmic operator to Eqn. (3.4). There is:

$$\begin{aligned} \max P(l(\boldsymbol{y}) \in l_X) &= \max \ln(P(l(\boldsymbol{y}) \in l_X)) \\ &= \min_{\alpha} \kappa \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + c\|\boldsymbol{\alpha}\|_2^2 \\ &= \min_{\alpha} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 \end{aligned} \tag{3.5}$$

where $\lambda = c/\kappa$. The above Eqn. (3.5) gives a probabilistic representation of $\boldsymbol{y}$ over the collaborative subspace $\mathcal{S}$. Interestingly, Eqn. (3.5) shares the same formulation of the representation formula of CRC [198], but it has a clear probabilistic interpretation.

3.3 Probabilistic Collaborative Representation

Our formulation in Section 3.2 provides a way to estimate the probability of $l(\boldsymbol{y}) \in l_X$ with the collaborative subspace $\mathcal{S}$. However, it cannot indicate which specific class k the sample $\boldsymbol{y}$ belongs. To perform classification, SRC/CRC simply uses the reconstruction error of $\boldsymbol{y}$ for each class-specific subspace to determine the class label. This classification rule is heuristic and lacks sufficient interpretation. Based on the proposed probabilistic collaborative subspace, in this section we present a probabilistic collaborative representation based classification to classify $\boldsymbol{y}$.

3.3.1 Probability to Each Class-specific Subspace

A sample $\boldsymbol{x} \in \mathcal{S}$ can be collaboratively represented as: $\boldsymbol{x} = \boldsymbol{X}\boldsymbol{\alpha} = \sum_{k=1}^{K} \boldsymbol{X}_k\boldsymbol{\alpha}_k$, where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1; \boldsymbol{\alpha}_2; \ldots; \boldsymbol{\alpha}_K]$ and $\boldsymbol{\alpha}_k$ is the coding vector associated with $\boldsymbol{X}_k$. Note that $\boldsymbol{x}_k = \boldsymbol{X}_k\boldsymbol{\alpha}_k$ is a data point falling into the subspace of class k. Again by using the Gaussian kernel, the probability that $\boldsymbol{x}$ has the same class label as $\boldsymbol{x}_k$ can be defined as

$$P(l(\boldsymbol{x}) = k | l(\boldsymbol{x}) \in l_X) \propto \exp(-\delta \|\boldsymbol{x} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2) \tag{3.6}$$

where δ is a constant.

For a query sample $\boldsymbol{y}$ outside the space $\mathcal{S}$, we can compute the probability that $l(\boldsymbol{y}) = k$ as:

$$\begin{aligned} P(l(\boldsymbol{y}) = k) &= P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) = k) \cdot P(l(\boldsymbol{x}) = k) \\ &= P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) = k) \cdot P(l(\boldsymbol{x}) = k | l(\boldsymbol{x}) \in l_X) \cdot P(l(\boldsymbol{x}) \in l_X). \end{aligned} \tag{3.7}$$

Since the probability definition in Eqn. (3.3) is independent of k as long as $k \in l_X$, we have $P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) = k) = P(l(\boldsymbol{y}) = l(\boldsymbol{x}) | l(\boldsymbol{x}) \in l_X)$. With Eqn. (3.5)~Eqn. (3.7), we have

$$\begin{aligned} P(l(\boldsymbol{y}) = k) &= P(l(\boldsymbol{y}) \in l_X) \cdot P(l(\boldsymbol{x}) = k | l(\boldsymbol{x}) \in l_X) \\ &\propto \exp(-(\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 + \gamma\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2)), \end{aligned} \tag{3.8}$$

where $\gamma = \delta/\kappa$.

3.3.2 The ProCRC Model

By maximizing the probability defined in Eqn. (3.8), we can find some data point $\boldsymbol{x}$ inside $\mathcal{S}$ (or equivalently the representation vector $\boldsymbol{\alpha}$) such that $P(l(\boldsymbol{y}) = k)$ achieves its maximum. However, if we maximize $P(l(\boldsymbol{y}) = k)$ individually for each class k, their corresponding data point $\boldsymbol{x}$ will be different. This makes the classification by the maximal $P(l(\boldsymbol{y}) = k)$ (w.r.t. k) unstable and less discriminative.

Alternatively, a better strategy is that we find a common data point $\boldsymbol{x}$ inside $\mathcal{S}$, which could maximize the joint probability $P(l(\boldsymbol{y}) = 1, \ldots, l(\boldsymbol{y}) = K)$. Once the common $\boldsymbol{x}$ is found, we can then check which probability $P(l(\boldsymbol{y}) = k)$ is the highest to determine the class label of $\boldsymbol{y}$. By assuming that the events $l(\boldsymbol{y}) = k$ are independent, we have

$$\begin{aligned} & \max P(l(\boldsymbol{y}) = 1, \ldots, l(\boldsymbol{y}) = K) = \max \prod\nolimits_k P(l(\boldsymbol{y}) = k) \\ \propto \quad & \max \exp(-(\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 + \frac{\gamma}{K}\sum\nolimits_{i=1}^{K}(\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_i\boldsymbol{\alpha}_i\|_2^2))). \end{aligned} \tag{3.9}$$

Applying the logarithmic operator to Eqn. (3.9) and ignoring the constant term, we have:

$$(\hat{\boldsymbol{\alpha}}) = \arg\min_{\alpha}\{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 + \frac{\gamma}{K}\sum\nolimits_{k=1}^{K}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2\}. \tag{3.10}$$

In Eqn. (3.10), the first two terms $\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2$ form a collaborative representation term, which encourages to find a point $\boldsymbol{x} = \boldsymbol{X}\boldsymbol{\alpha}$ that is close to $\boldsymbol{y}$ in the collaborative subspace $\mathcal{S}$. The last term $\sum_{k=1}^{K}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2$ attempts to find inside each subspace of class k a point $\boldsymbol{X}_k\boldsymbol{\alpha}_k$ which is close to the common point $\boldsymbol{x}$. The parameters γ and λ balance the role of the three terms, which can be set based on the prior knowledge of the problem, or we can use the cross-validation technique to determine γ and λ from the training data. When the regularization parameter $\gamma = 0$, Eqn. (3.10) will degenerate to CRC, and the term $\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2$ will play a dominant role in determining $\boldsymbol{\alpha}$. When the regularization parameter $\gamma > 0$, the term $\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2$ is introduced to further adjust $\boldsymbol{\alpha}_k$ by $\boldsymbol{X}_k$, which results in a more robust and stable solution to $\boldsymbol{\alpha}$.

3.3.3 The ProCRC Classifier

With the model in Eqn. (3.10), a solution vector $\hat{\boldsymbol{\alpha}}$ is obtained. The probability $P(l(\boldsymbol{y}) = k)$ can be computed by:

$$P(l(\boldsymbol{y}) = k) \propto \exp(-(\|\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\alpha}}\|_2^2 + \lambda\|\hat{\boldsymbol{\alpha}}\|_2^2 + \frac{\gamma}{K}\|\boldsymbol{X}\hat{\boldsymbol{\alpha}} - \boldsymbol{X}_k\hat{\boldsymbol{\alpha}}_k\|_2^2)). \tag{3.11}$$

Note that $(\|\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\alpha}}\|_2^2 + \lambda\|\hat{\boldsymbol{\alpha}}\|_2^2)$ is the same for all classes, and thus we can omit it in computing $P(l(\boldsymbol{y}) = k)$. Let

$$p_k = \exp(-(\|\boldsymbol{X}\hat{\boldsymbol{\alpha}} - \boldsymbol{X}_k\hat{\boldsymbol{\alpha}}_k\|_2^2)). \tag{3.12}$$

The classification rule can then be formulated as

$$l(\boldsymbol{y}) = \arg\max_k\{p_k\}. \tag{3.13}$$

We call the above classifier probabilistic collaborative representation based classifier (ProCRC).

3.3.4 The Robust ProCRC Model

In visual classification, partial corruption or occlusion often degrade the performance. It is well-known that the robustness of classification tasks can be enhanced by using ℓ_1-norm to characterize the loss function [177]. Our proposed probabilistic collaborative representation in Section 3.2.2 can be easily extended to its robust version. In Eqn. (3.3), we can choose to use the Laplacian kernel, instead of the Gaussian kernel, to measure the probability:

$$P(l(\boldsymbol{y}) = l(\boldsymbol{x})|l(\boldsymbol{x}) \in l_X) \propto \exp(-\kappa\|\boldsymbol{y} - \boldsymbol{x}\|_1). \tag{3.14}$$

With similar derivations to ProCRC, we can have the following robust ProCRC (R-ProCRC) model:

$$(\hat{\boldsymbol{\alpha}}) = \arg\min_{\boldsymbol{\alpha}}\{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\alpha}\|_1 + \lambda\|\boldsymbol{\alpha}\|_2^2 + \frac{\gamma}{K}\sum_{k=1}^{K}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2\}. \tag{3.15}$$

The classification rule is the same as that in Eqn. (3.13).

3.3.5 Solutions to ProCRC and R-ProCRC Models

The proposed ProCRC model has a closed form solution, while the proposed R-ProCRC model can be easily solved by the iterative reweighted least square (IRLS) technique.

ProCRC. Refer to Eqn. (3.15), let $\boldsymbol{X}'_k$ be a matrix which has the same size as $\boldsymbol{X}$, while only the samples of $\boldsymbol{X}_k$ will be assigned to $\boldsymbol{X}'_k$ at their corresponding locations in $\boldsymbol{X}$, *i.e.*, $\boldsymbol{X}'_k = [\boldsymbol{0}, \ldots, \boldsymbol{X}_k, \ldots, \boldsymbol{0}]$. Let $\overline{\boldsymbol{X}}'_k = \boldsymbol{X} - \boldsymbol{X}'_k$. We can then compute the following projection matrix offline:

$$\boldsymbol{T} = (\boldsymbol{X}^T\boldsymbol{X} + \frac{\gamma}{K}\sum\nolimits_{k=1}^{K}(\overline{\boldsymbol{X}}'_k)^T\overline{\boldsymbol{X}}'_k + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T, \tag{3.16}$$

where $\boldsymbol{I}$ denotes the identity matrix. With $\boldsymbol{T}$, the solution to $\boldsymbol{\alpha}$ can be obtained efficiently:

$$\hat{\boldsymbol{\alpha}} = \boldsymbol{T}\boldsymbol{y}. \tag{3.17}$$

R-ProCRC. Though the proposed R-ProCRC model is convex, there is no closed form solution to it, and we adopt an IRLS algorithm to compute $\boldsymbol{\alpha}$.

Based on the current estimation of $\boldsymbol{\alpha}$, we introduce the diagonal weighting matrix $\boldsymbol{W}_x$:

$$\boldsymbol{W}_x(i,i) = 1/|\boldsymbol{X}(i,:)\boldsymbol{\alpha} - \boldsymbol{y}_i|, \tag{3.18}$$

where $\boldsymbol{X}(i,:)$ refers to the ith row of $\boldsymbol{X}$. Given $\boldsymbol{W}_x$, the problem in Eqn. (3.15) can be reformulated as:

$$(\hat{\boldsymbol{\alpha}}) = \arg\min_{\alpha}\{\frac{\gamma}{K}\sum\nolimits_{k=1}^{K}\|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{X}_k\boldsymbol{\alpha}_k\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 + (\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y})^T\boldsymbol{W}_x(\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y})\}. \tag{3.19}$$

Then the coefficient vector $\boldsymbol{\alpha}$ can be updated by:

$$\hat{\boldsymbol{\alpha}} = (\boldsymbol{X}^T\boldsymbol{W}_x\boldsymbol{X} + \frac{\gamma}{K}\sum\nolimits_{k=1}^{K}(\overline{\boldsymbol{X}}'_k)^T\overline{\boldsymbol{X}}'_k + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{W}_x\boldsymbol{y}. \tag{3.20}$$

We alternatively update the weighting matrices $\boldsymbol{W}_x$ and the coefficient vector $\boldsymbol{\alpha}$, and stop until convergence or after a fixed number of iterations.

3.4 Experimental Results

In this section, we comprehensively evaluate the proposed method from different aspects. In 3.4.1, by using the (MNIST [65] and USPS [65]) datasets, we compare ProCRC with state-of-the-art representation based classifiers along this line, including NSC [22], SRC [177], CRC [198] and CROC [20, 21]. The linear support vector machine (SVM) classifier [34] is also compared. In 3.4.2, we compare R-ProCRC with robust SRC [186] on robust face recognition using the AR [105] and Extended Yale B [44] datasets. In 3.4.3, we evaluate the running time of ProCRC. Finally, in 3.4.4 we evaluate ProCRC on several challenging visual classification datasets, including Stanford 40 Actions dataset [189], Caltech-UCSD Birds-200-2011 dataset [167], Oxford 102 Flowers dataset [118], Caltech-256 dataset [53] and ImageNet ILSVRC 2012 dataset [135].

The proposed ProCRC has two parameters, λ and γ. In the experiments, we set $\lambda = 10^{-3}$ for handwritten digit datasets and face datasets, and $\lambda = 10^{-2}$ for other datasets. For the parameter γ, we set it by 5-fold cross-validation on the training set. For those competing classifiers, their source codes are from the original authors, and we tune their parameters to achieve their best classification accuracy in each experiment.

3.4.1 Handwritten Digit Recognition

MNIST dataset: The MNIST [65] dataset contains a training set of 60,000 samples and a test set of 10,000 samples. There are 10 classes, and the size of each image is 28×28. We randomly selected 50, 100, 300, and 500 samples from each class for training, and we used all the samples in the test set for testing.

USPS dataset: The USPS [65] dataset also contains a training sample set and a test sample set, and the size of each image is 16×16. We randomly selected 50, 100, 200, and 300 samples from each class for training, and used all the samples in the test set for testing.

Table 3.1 Classification rate (%) on the MNIST dataset.

Num.	50	100	300	500
SVM	89.35	92.10	94.88	**95.93**
NSC	91.06	92.86	85.29	78.26
CRC	72.21	82.22	86.54	87.46
SRC	80.12	85.63	89.30	92.70
CROC	91.06	92.86	89.93	89.37
ProCRC	**91.84**	**94.00**	**95.48**	95.88

Table 3.2 Classification rate (%) on the USPS dataset.

Num.	50	100	200	300
SVM	93.46	95.31	95.91	96.30
NSC	93.48	93.25	90.21	87.85
CRC	89.89	91.67	92.36	92.79
SRC	92.58	93.99	95.63	95.86
CROC	93.48	93.25	91.40	91.87
ProCRC	**93.84**	**95.62**	**96.03**	**96.43**

Table 3.1 and Table 3.2 list the classification rates on the two datasets, respectively. We can see that ProCRC outperforms all the competing classifiers. With the increase of the number of training samples, the classification accuracy of ProCRC increases consistently; however, the classification rate of NSC drops with the increase of training samples, while the rate of CRC first jumps and then increases a little. This shows that ProCRC has the good robustness to the number of training samples by considering all the classes collaboratively while double checking individual class. It has the smallest performance variation under the different number of training samples.

3.4.2 Face Recognition with Corruption

We then evaluate R-ProCRC for face recognition (FR) with partial occlusion or corruption. The AR [105] and Extended Yale B [44] datasets are used since they are commonly used to in the original papers to evaluate SRC, CRC and CROC. Three types of corruptions are considered: random pixel corruption, random block occlusion, and disguise. In the experiments of random pixel corruption, for each test image, we randomly select a certain percentage of pixels and replace them with uniformly distributed values within $[0, 255]$. In the experiments of block occlusion, for each test image, we randomly select a square block and replace it with an unrelated image. For real disguise, we use the images with sunglasses or scarf in the AR dataset.

Since the SVM, NSC, CRC and CROC classifiers do not consider the robustness to outliers in design, we only compare R-ProCRC with the robust version (ℓ_1-norm loss function and regularizer) of SRC, denoted by R-SRC [186].

Random corruption: We use the Extended Yale B dataset to evaluate R-ProCRC against random corruption. We randomly selected 30 images from each subject to construct the training dataset, and used the remaining images for testing. Random corruption is added to each test image. Table 3.3 lists the recognition rates of R-SRC and R-ProCRCr under different ratios of random corruption. One can see that R-ProCRC is much better than R-SRC for FR with random corruption.

Block occlusion: We then compare R-SRC with R-ProCRC for FR with block occlusion. The same experiment setting as in the random corruption experiment is used by changing random corruption to random corruption. The results are listed in Table 3.4. One can see that block occlusion will cause more significant performance degradation than random corruption, while R-ProCRC still significantly outperforms R-SRC under different ratios of block occlusion.

Disguise: At last, we use the face images with the disguise in the AR dataset to evaluate

R-ProCRC. We used the 700 non-occluded images in the first session for training, and used the 600 images with sunglasses and the 600 images with the scarf for testing. Table 3.5 lists the experimental results. Again, R-ProCRC is consistently superior to R-SRC.

Table 3.3 Recognition rate (%) on face images with random corruption on the Extended Yale B dataset.

Corruption ratio	10%	20%	40%	60%
R-SRC [186]	97.49	95.60	90.19	76.85
R-ProCRC	**98.45**	**98.20**	**93.25**	**82.42**

Table 3.4 Recognition rate (%) on face images with block occlusion on the Extended Yale B dataset.

Corruption ratio	10%	20%	30%	40%
R-SRC [186]	90.42	85.64	78.89	70.09
R-ProCRC	**98.12**	**92.62**	**86.42**	**77.16**

Table 3.5 Recognition rate (%) on face images with disguise on the AR dataset.

Corruption ratio	Sunglasses	Scarf
R-SRC [186]	69.17	69.50
R-ProCRC	**70.50**	**69.83**

3.4.3 Running Time Comparison

We evaluate the running time of ProCRC and the competing representation based classifiers by processing one test image on the MNIST dataset (5,000 samples for training), and evaluate the running time of R-ProCRC and R-SRC by processing one image on the AR dataset (we test the disguise problem on 600 images with scarf). All methods are implemented in Matlab,

and run on a PC with Intel (R) Core (TM) i7-5930K 3.50 GHz CPU and 32 GB RAM. Table 3.6 lists the running time of different methods.

Since ProCRC and CRC have analytical solutions and the resolved projection matrices have the same size, they have the same speed, which is faster than CROC and much faster than SRC. R-ProCRC employs ℓ_1-norm only for loss function, while R-SRC employs ℓ_1-norm for both loss and regularization. Therefore, R-ProCRC is faster than R-SRC.

Table 3.6 Running time (s) of different methods.

Method	NSC	CRC	SRC	CROC
Time (s)	0.0003	0.0005	0.22	0.0009
Method	ProCRC	R-SRC	R-ProCRC	
Time (s)	0.0005	3.57	1.81	

3.4.4 Other Challenging Visual Classification Tasks

Datasets and settings. To more comprehensively assess the performance of ProCRC, we apply it to four challenging classification datasets: Stanford 40 Actions dataset [189] for action recognition, Caltech-UCSD Birds-200-2011 [167] and Oxford 102 Flowers datasets [118] for fine-grained object recognition, and Caltech-256 dataset [53] for large-scale object recognition. We do not evaluate R-ProCRC since corruption is not the main problem in these datasets.

Stanford 40 Actions dataset [189] is composed of 40 human actions, *e.g.*, brushing teeth, cleaning the floor, reading book, throwing a Frisbee. It contains 9352 images, with 180~300 images per class. We follow the training-test split settings suggested by the authors [189], using 100 images from each class for training and the remaining for testing.

Caltech-UCSD Birds-200-2011 (CUB) dataset [167] is a widely-used benchmark for fine-grained image recognition, which contains 11,788 images of 200 bird species. Due to

the high degree of similarity among species, this dataset is very challenging. We used the split setting provided in the dataset without part or bounding box annotations. There are around 30 training samples for each species.

Oxford 102 Flowers dataset [118] is another fine-grained image classification benchmark which contains 8,189 images from 102 categories, and each category has at least 40 images. The flowers appear at different scales, pose and lighting conditions. This dataset is challenging since there exist large variations within the category but small difference across several categories.

Caltech-256 dataset [53] consists of 256 object categories with at least 80 images per category. This dataset has a total number of 30,608 images. Following the common experimental settings, we randomly selected 15, 30, 45 and 60 images from each category for training, respectively, and used the remaining images for testing. For a fair comparison, we run ProCRC 10 times for each partition and report the average classification accuracy.

On the four datasets, we employ two types of features to demonstrate the effectiveness of ProCRC. First, we use VLFeat [165] to extract the BOW feature based on SIFT (refer to BOW-SIFT feature). The square patch size and stride are set at 16×16 and 8 pixels, respectively. The codebook is trained by the k-means method, and the size is 1,024. We use a 2-level spatial pyramid representation. The final feature dimension of each image is 5,120 for all datasets. Second, we use VGG-verydeep-19 [146] to extract CNN features (refer to VGG19 features). We use the activations of the penultimate layer as local features, which are extracted from 5 scales $\{2^s, s = -1, -0.5, 0, 0.5, 1\}$. We pool all local features together regardless of scales and locations. The final feature dimension of each image is 4,096 for all datasets. Both BOW-SIFT and VGG19 features are ℓ_2 normalized.

Evaluation of different classifiers with the BOW-SIFT features and CNN feature. To verify that ProCRC is an effective classifier, we present a detailed comparison between ProCRC and several widely-used classifiers, including softmax, linear SVM, kernel SVM with

Table 3.7 Accuracies (%) of different classifiers with BOW-SIFT features and VGG19 features.

Classifier	Standford 40		CUB		Flower 102		Caltech 256	
	BOW-SIFT	VGG19	BOW-SIFT	VGG19	BOW-SIFT	VGG19	BOW-SIFT(30)	VGG19(30)
Softmax	21.1	77.2	8.2	72.1	46.5	87.3	25.8	75.3
SVM	24.0	79.0	10.2	75.4	50.1	90.9	28.5	80.1
Kernel SVM	26.3	79.8	**10.5**	76.6	51.0	92.2	28.7	81.3
NSC	22.1	74.7	8.4	74.5	46.7	90.1	25.8	80.2
CRC	24.6	78.2	9.4	76.2	49.9	93.0	27.4	81.1
SRC	24.2	78.7	7.7	76.0	47.2	93.2	26.9	81.3
CROC	24.5	79.1	9.1	76.2	49.4	93.1	27.9	81.7
ProCRC	**28.4**	**80.9**	9.9	**78.3**	**51.2**	**94.8**	**29.6**	**83.3**

χ^2 kernel, CRC, SRC and CORC. The classification rates on the four datasets with BOW-SIFT features and VGG19 features are listed in Table 3.7 (the results on the Caltech-256 dataset are obtained by using 30 training images per category). From Table 3.7, we can see that ProCRC almost always achieves the best accuracy with either BOW-SIFT features or VGG19 features among all the classifiers. Specifically, with the powerful CNN features, ProCRC obtains at least 1.5% performance gains over all the other classifiers. These results clearly demonstrate the effectiveness of ProCRC as a visual classifier.

Comparison to state-of-the-art methods. Furthermore, we compare ProCRC (using the VGG19 features) with the state-of-the-art methods on each dataset in Table 3.8. Note that many of the comparison methods are CNN based methods and their features are even stronger than VGG19.

The classification accuracies on Standford 40 Actions dataset are from SPM [185], LLC [170], EPM [143], SparseBases [189], CF [77], SMP [78] and ASPD [141]. We see that ProCRC achieves at least 5.5% improvement over others. As can be seen in Table 3.7, using the same VGG19 features, kernel SVM leads to an accuracy of 79.8%, which is 1.1% lower than ProCRC.

The classification accuracies on CUB dataset are from POOF [8], FV-CNN [25], PN-CNN [11] and NAC [144]. Again, ProCRC outperforms all methods except for NAC. However,

Table 3.8 Comparsions to state-of-the-arts on different datasets (Standford 40, CUB, Flower 102 and Caltech-256).

Dataset	Split	Methods & Accuracies (%)							
Standford 40	fixed	ProCRC	ASPD	SMP	CF	SparBases	EPM	LLC	ScSPM
		80.9	75.4	53.0	51.9	45.7	42.2	35.2	34.9
CUB	fixed	ProCRC	NAC	PN-CNN	FV-CNN	POOF			
		78.3	**81.0**	75.7	66.7	56.9			
Flower 102	fixed	ProCRC	NAC	OverFeat	GMP	DAS	BiCos seg		
		94.8	**95.3**	86.8	84.6	80.7	79.4		
Caltech-256	random	ProCRC	NAC	VGG19	CNN-S	ZF	M-HMP	LLC	ScSPM
	15	**80.2**	-	-	-	65.7	42.7	34.4	27.7
	30	**83.3**	-	-	-	70.6	50.7	41.2	34.0
	45	**84.9**	-	-	-	72.7	54.8	45.3	37.5
	60	**86.1**	84.1	85.1	77.6	74.2	58.0	-	-

please note that NAC further constructs a part-model based on the VGG19 feature for recognition, while ProCRC performs classification directly using the VGG19 feature. Compared with the other three methods which all use a specially designed CNN architecture for bird species recognition, the improvement by ProCRC is obvious.

The classification accuracies on Oxford 102 Flowers dataset are from BiCos seg [17], DAS [4], GMP [115], OverFeat [132] and NAC [144]. ProCRC improves 8% over OverFeat and is only 0.5% lower than NAC, which uses an additional part-model VGG19 feature. The performance gain is significant compared with BiCos seg, DAS and GMP (increase by 15.4%, 14.1% and 10.2%, respectively).

The average classification accuracies (over 10 runs) on Caltech-256 dataset are from ScSPM [185], LLC [170], M-HMP [9], ZF [191], CNN-S [19], VGG19 [146] and NAC [144]. The symbol "-" means that the result is not reported in the original work. ProCRC has at least 12% performance gain over ZF, and has more significant improvements over ScSPM, LLC, M-HMP. When 60 images per class are used for training, ProCRC achieves 1% improvement compared with VGG19 + linear SVM (85.1%), and 2% improvement compared with NAC,

while the latter even uses an additional part-model based on the VGG19 feature.

The scalability of ProCRC. In the proposed ProCRC model, a matrix inversion operation (see Eqn. (3.16) will be involved to obtain the projection matrix $\boldsymbol{T}$. The dimensionality of this matrix inverse depends on the number of training samples in the dataset. Therefore, one potential problem of ProCRC is its scalability on very large scale datasets which have millions of training samples (*e.g.*, ImageNet [135]). It might not be feasible to load millions of samples into memory and solve a matrix inverse problem with the dimensionality of millions.

Fortunately, the scalability problem of ProCRC can be solved by using the dictionary learning (DL) techniques. More specifically, for a dataset which has a large number of samples per class, we can learn a compact dictionary $\boldsymbol{D}_k$, which has only a small number of atoms, from the original samples $\boldsymbol{X}_k$. The ProCRC classifier can then be applied by replacing $\boldsymbol{X}_k$ by $\boldsymbol{D}_k$. One simple DL model is $\min_{\{\boldsymbol{D}_k,\boldsymbol{A}_k\}} \|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2 + \tau\|\boldsymbol{A}_k\|_F^2$, where τ is a trade-off parameter and each column of $\boldsymbol{D}_k$ has unit length. This DL model can be easily solved by using an alternating optimization procedure to update $\boldsymbol{D}_k$ and $\boldsymbol{A}_k$.

With the above mentioned DL strategy, we test ProCRC (and other representation based classifiers) on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 dataset [135], which consists of 1.2M+ training images from 1,000 categories (about 1,300 images per category) and 50K validation images (50 images per category). We compare ProCRC with other classifiers using two baseline visual features: BOW-SIFT extracted by VLFeat (we use a codebook of 1,000 visual words to perform the k-means method, and the feature dimension is 1,000 since 0-level spatial pyramid representation is adopted here for simplicity) and AlexNet features extracted by Caffe (as described in [84], the feature dimension is 4,096). For each category, a dictionary with 50 atoms is learned from the about 1,300 samples.

The top-1 and top-5 classification accuracies are listed in Table 3.9. With the handcraft BOW-SIFT feature, the top-1 accuracy of ProCRC is at least 1.1% higher than all the other competitive classifiers. With the AlexNet based CNN feature, ProCRC outperforms SVM

Table 3.9 Accuracies (%) on ImageNet ILSVRC-2012.

Classifier	BOW-SIFT		AlexNet	
	top-5	top-1	top-5	top-1
Softmax	28.8	7.4	**80.4**	**57.4**
SVM	29.1	7.2	79.7	55.8
NSC	27.4	6.6	77.4	53.2
CRC	28.3	7.3	78.5	54.3
SRC	28.6	6.9	78.7	54.1
CROC	28.5	7.2	78.8	54.4
ProCRC	**29.7**	**8.5**	80.1	56.3

(0.5%) and other representation based classifiers (2%), but is 1.1% and 0.3% lower than Softmax on top-1 and top-5 accuracies, respectively. This is mainly because of the fact that AlexNet features are trained with the Softmax output layer. In summary, DL is capable to solve the scalability issue of ProCRC. In the future, we will explore other methods (*e.g.*, a hierarchical structure) to further improve the performance and scalability of ProCRC.

3.5 Conclusion

We presented a probabilistic collaborative representation based classifier, namely ProCRC, which employs a probabilistic collaborative representation framework to jointly maximize the probability that a test sample belongs to each class. ProCRC effectively makes use of the training samples from all classes to deduce the class label of a test sample. It possesses a clear probabilistic interpretation, and is very efficient to solve. Our experiments on handwritten digit recognition, face recognition, and other visual classification tasks validated its superiority to popular representation based classifiers, including NSC, CRC, SRC and CROC, as well as benchmark classifiers such as SVM and kernel SVM. Coupled with CNN features (*e.g.*,

VGG19), ProCRC demonstrated state-of-the-art performance on challenging visual datasets such as Stanford 40 Actions, CUB, Oxford 102 Flowers, and Caltech-256. We also demonstrated that ProCRC could be applied to larger-scale dataset such as ImageNet ILSVRC-2012 by introducing a simple dictionary learning pre-processing stage.

Chapter 4

Higher-order Integration of Hierarchical Convolutional Activations for Fine-grained Visual Categorization

4.1 Introduction

Deep CNNs have emerged as the new state-of-the-art for a wide range of visual recognition tasks. Nevertheless, it remains quite challenging to derive the effective discriminative representation for FGVC, primarily due to subtle semantic differences between subordinate categories. Conventional CNNs usually deploy the fully connected layers to learn global semantic representation and may not be suitable to FGVC. Therefore, leveraging local discriminative patterns in CNN is crucial to obtain a more powerful representation, and recently has been intensively studied for FGVC.

Part-based representations [11, 144, 193, 199, 201] built on CNN features have been a predominant trend in FGVC. Such methods follow a detection module consisting of part detection and appearance modelling to extract regional features on deeper convolutional layers

in R-CNN [45] based scenario. Then global appearance structure is incorporated to pool these regional features. Although these methods have yielded rich empirical returns, they still pose the following issues: (1) A considerable number of part-based methods [11, 193, 199] heavily rely on the detailed part annotations to train accurate part detectors, which is costly and further limits the scalability for large-scale datasets; moreover, identifying discriminative parts for specific fine-grained objects is quite challenging and often requires interaction with human or expert knowledge [12, 168]; (2) The discriminative semantic parts in images often appear at different scales. As each spatial unit in the deeper convolutional layer corresponds to a specific receptive field, activations from a single convolutional layer are limited in describing various parts with different sizes; (3) Exploiting the joint configuration of individual object parts is very important for object appearance modelling. A few works introduce additional geometric constraints for object parts including the popular deformable parts model [199], constellation model [144] and order-shape constraint [173]. One key disadvantage of these approaches is that they only characterize the first-order occurrences and relationships of very few parts, however, cannot be readily applied to model objects with more parts. Consequently, our focus is to capture the higher-order statistics of those semantic parts at different scales, and thus provide a more flexible way for global appearance modelling without the help of part annotation.

In recent works [144, 201], the deeper convolutional filters are regarded as weak part detectors and the corresponding activations as the responses of detection, as shown in Fig. 4.1. Motivated by this observation, instead of part annotations and explicit appearance modelling, we straightforwardly exploit the higher-order statistics from the convolutional activations. We first provide a perspective of the matching kernel to understand the widely adopted mapping and pooling schemes on convolutional activations in conjunction with a linear classifier. Linear mapping and direct pooling only capture the occurrence of parts. In order to capture the higher-order relations among parts, it is better to explore local non-linear matching kernels to characterize the higher-order part interactions (*e.g.*, co-occurrence). However, designing an

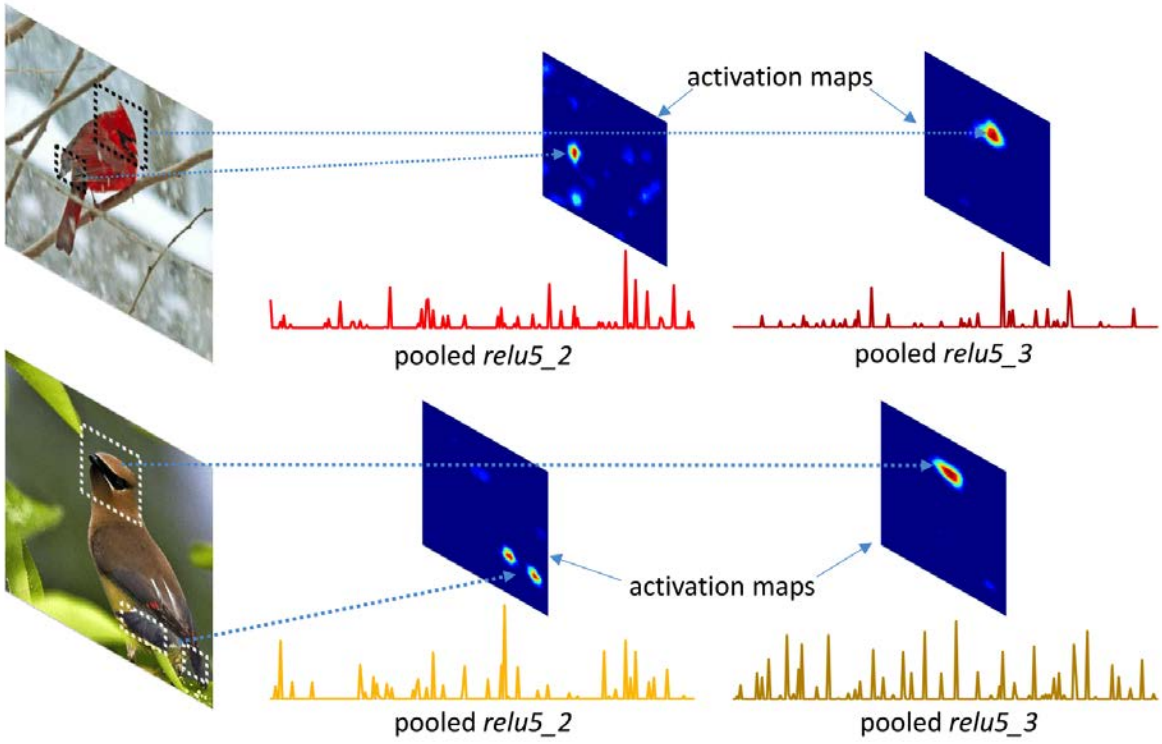


Figure 4.1 Visualization of several activation maps that corresponds to large responses of the sum-pooled vectors of two activation layers *relu5_2* and *relu5_3* in VGG-16 model.

appropriate CNN architecture that can be plugged with non-linear local kernels in an end-to-end manner is non-trivial. The kernel scheme is required to have explicit non-linear maps and be differentiable to facilitate back-propagation. One representative work is convolutional kernel network (CKN) [103], which provides a kernel approximation scheme to interpret CNNs. A related polynomial network [98] is to utilize polynomial activation functions as alternatives of ReLU in CNNs to learn non-linear interactions of feature variables. Similarly, we leverage the polynomial kernel to serve in modelling higher-level part interactions and derive the polynomial modules that allow trainable structure built on CNNs.

With the kernel scheme, we extend our framework for higher-order integration of hierarchical convolutional activations. The effectiveness of fusing hierarchical features in CNNs has been widely reported in visual recognition. The benefits come from both the different discrim-

inative capacities of multiple convolutional layers and the coarse-to-fine object description. However, the existing methods merely concatenate or sum multiple activations into a holistic representation [60], or adopt a decision level fusion to combine side-outputs from different layers [90, 179]. These methods, however, are limited in exploiting the intrinsic higher-order relationships of convolutional activations in either the intra-layer level or the inter-layer level. By using the kernel fusion on hierarchical convolutional activations, we can construct a richer image representation for cross-layer integration. Compared with the related works that perform feature fusion via learning multiple networks [26, 94, 145], our framework is easy to construct and more effective for FGVC.

4.2 Related Work

4.2.1 Feature Encoding in CNNs

Applying encoding techniques for the local convolutional activations in CNNs has shown significant improvements compared with the fully-connected outputs [25, 182]. In this case, the VLAD and FV as high-order statistics based representation can be readily applied. Gong *et al.* [47] propose to use VLAD to encode local features extracted from multiple regions of an image. In [25, 29, 182], the values of FV encoding on convolutional activations are discovered for scene, texture and video recognition tasks. However, regarding feature encoding as an isolated component is not the optimal choice for CNNs. Therefore, Lin *et al.* [94] propose a bilinear CNN (B-CNN) as codebook-free coding that allows end-to-end training for FGVC. The very recent work in [5] builds a weakly place recognition system by introducing a generalized VLAD layer that can be trained with off-the-shelf CNN models. An alternative for feature mapping is to exploit kernel approximation feature embedding. Yang *et al.* [188] introduce adaptive Fastfood transform in their deep fried convnets to replace the fully-connected layers, which is a generalization of the Fastfood transform for approximating kernels [89].

Gao *et al.* [42] implement an end-to-end structure to approximate the degree-2 homogeneous polynomial kernel by utilizing random features and sketch techniques.

4.2.2 Feature Fusion in CNNs

Compared with the fully connected layers capturing the global semantic information, convolutional layers preserve more instance-level details and exhibit diverse visual contents as well as different discriminative capacities, which are more meaningful to the fine-grained recognition task [6]. Recently a few works attempt to investigate the effectiveness of exploiting features from different convolutional layers [95, 183]. Long *et al.* [99] combine the feature maps from intermediate level and high level convolutional layers in their fully convolutional network to provide both finer details and higher-level semantics for better image segmentation. Hariharan *et al.* [60] introduce hypercolumns for localization and segmentation, where convolutional activations at a pixel of different feature maps are concatenated as a vector as a pixel descriptor. Similarly, Xie and Tu [179] present a holistically-nested edge detection scheme in which the sideoutputs are added after several lower convolutional layers to provide deep supervision for predicting edges at multiple scales.

4.3 Kernelized Convolutional Activations

Most part-based CNN methods for FGVC consist of two components: (i) feature extraction for semantic parts on the last convolutional layer, and (ii) spatial configuration modelling for those parts to produce the discriminative image representation. In this work, we treat the convolutional filter as the part detector, and then the convolutional activations in a single spatial position can be considered as the part descriptions. Therefore, instead of explicit part extraction, we introduce polynomial predictor to integrate a family of local matching kernels for modelling higher-order part interactions and derive powerful representation for FGVC.

4.3.1 Matching Kernel and Polynomial Predictor

Suppose that an image I is passed by a plain CNN, and we denote the 3D activations $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{K\times M\times N}$ extracted from some specific convolutional layer as a set of K-dimensional descriptors $\{\boldsymbol{x}_p\}_{p\in\Omega}$, where K is the number of feature channels, $\boldsymbol{x}_p$ represents the descriptor at a particular position p over the set Ω of valid spatial locations ($|\Omega| = M \times N$). We first consider the matching scheme $\mathcal{K}$ for activation sets $\boldsymbol{\mathcal{X}}$ and $\bar{\boldsymbol{\mathcal{X}}}$ from two images, in which the set similarity is measured via aggregating all the pairwise similarities among the local descriptors:

$$\mathcal{K}(\boldsymbol{\mathcal{X}}, \bar{\boldsymbol{\mathcal{X}}}) = Agg(\{k(\boldsymbol{x}_p, \bar{\boldsymbol{x}}_{\bar{p}})\}_{p\in\Omega,\bar{p}\in\bar{\Omega}}) = \psi(\boldsymbol{\mathcal{X}})^T\psi(\bar{\boldsymbol{\mathcal{X}}}), \tag{4.1}$$

where $k(\cdot)$ is some kernel function between individual descriptors of two activation sets, $Agg(\cdot)$ is some set-based aggregation function, $\psi(\boldsymbol{\mathcal{X}})$ and $\psi(\bar{\boldsymbol{\mathcal{X}}})$ are the vector represetations for sets. It is worth noting that the construction of $\mathcal{K}$ presented above is decomposed into two steps in CNNs: feature mapping and feature aggregation. The mapping step maps each local descriptor $\boldsymbol{x} \in \mathbb{R}^K$ as $\phi(\boldsymbol{x}) \in \mathbb{R}^D$ in elaborated feature space. The aggregating step produces an image-level representation $\psi(\boldsymbol{\mathcal{X}})$ from the set $\{\phi(\boldsymbol{x}_p)\}_{p\in\Omega}$ through some pooling function $g(\cdot)$.

The key for FGVC is to discover and represent those local regions which share common appearances within the same category while exhibiting distinctive difference across categories. Based on the matching scheme $\mathcal{K}$ in Eqn. (4.1), appropriate pooling operators have been designed to efficiently prune non-discriminative matching subset while retaining those highly discriminative ones into image representation. Among them, sum pooling assigns equal weights to each position, and does not emphasize any position. Max pooling only considers the most significant position, which results in enormous information loss and is prone to small perturbation. Other pooling operators such as generalized max pooling [115] and ℓ_p-norm pooling [36] may be effective in discovering informative regions, but the feasible end-to-end schemes are unclear. Our attention is to model the higher-order relationships for discriminative representation of local patch and design suitable local mapping function ϕ which can

be stacked upon CNN for end-to-end training. Thus, we simply adopt $g(\cdot)$ as the global sum pooling, in which case we denote it as:

$$\psi(\boldsymbol{\mathcal{X}}) = g(\{\phi(\boldsymbol{x}_p)\}_{p\in\Omega}) = \sum_{p\in\Omega} \phi(\boldsymbol{x}_p). \tag{4.2}$$

The above matching underpinning highlights the advantage of generating image-level representation compatible with linear predictors, which can be interpreted as the linear combination of all local compositions accordingly:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}) \rangle, \tag{4.3}$$

where $\boldsymbol{w}$ is the parameter of predictor, we omit the bias term and position subscript p here for later convenience. As our aim is to capture more complex and higher-order relationships among parts, to this end, we propose the following polynomial predictor:

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} w_k x_k + \sum_{r=2}^{R} \sum_{k_1,\ldots,k_r} \mathcal{W}^r_{k_1,\ldots,k_r} (\prod_{s=1}^{r} x_{k_s}), \tag{4.4}$$

where R is the maximal degree of part interactions, $\boldsymbol{\mathcal{W}}^r$ is a r-order tensor which contains the weights of degree-r variable combinations in $\boldsymbol{x}$. For instance, when $r = 3$, $\mathcal{W}_{i,j,k}$ is the weight of $x_i x_j x_k$. We discuss different polynomial predictors as well as their corresponding kernels as follows:

1) **Linear kernel:** $k(\boldsymbol{x}, \bar{\boldsymbol{x}}) = \langle \boldsymbol{x}, \bar{\boldsymbol{x}} \rangle$ is the most simple kernel that refers to an identity map $\phi : \boldsymbol{x} \mapsto \boldsymbol{x}$, which is identical to the polynomial predictor of degree-1: $f(\boldsymbol{x}) = \sum_{k=1}^{K} w_k x_k$.

2) **Homogeneous polynomial kernel:** $k(\boldsymbol{x}, \bar{\boldsymbol{x}}) = \langle \boldsymbol{x}, \bar{\boldsymbol{x}} \rangle^r$ has shown the superiority in characterizing the intrinsic manifold structure of dense local descriptors [16]. The induced nonlinear map $\phi : \boldsymbol{x} \mapsto \otimes_r \boldsymbol{x}$, where $\otimes_r \boldsymbol{x}$ is a tensor defined by the r-order self-outer product [124] of $\boldsymbol{x}$, is able to model all the degree-r interactions between variables. Its polynomial predictor obeys the following form:

$$f(\boldsymbol{x}) = \sum_{k_1,\ldots,k_r} \mathcal{W}^r_{k_1,\ldots,k_r} (\prod_{s=1}^{r} x_{k_s}). \tag{4.5}$$

Notice that the polynomial predictor of degree-2 homogeneous polynomial kernel is defined as $\sum_{i,j} W_{i,j} x_i x_j$, which captures all pairwise/second-order interactions between variables and is an increasingly popular model in classification tasks [94].

3) **Positive definite kernel:** as discussed in [75], the positive definite kernel $k(\boldsymbol{x}, \bar{\boldsymbol{x}}) : (\boldsymbol{x}, \bar{\boldsymbol{x}}) \mapsto f(\langle \boldsymbol{x}, \bar{\boldsymbol{x}} \rangle)$ defines an analytic function which admits a Maclaurin expansion with only nonnegative coefficients, *i.e.*, $f(x) = \sum_{r=0}^{\infty} a_r x^r$, $a_r \geq 0$. For instance, a non-homogeneous degree-2 polynomial kernel $(\langle \boldsymbol{x}, \bar{\boldsymbol{x}} \rangle + 1)^2$ corresponds to a polynomial predictor that captures all single and pairwise interactions between variables. It also indicates that the positive definite kernel can be approximated arbitrarily accurately by polynomial kernels in principle of sufficiently high degree polynomial expansions for target functions.

4.3.2 Tensor Learning for Polynomial Kernels

Before deriving the end-to-end CNN architecture for learning the parameters in Eqn. (4.4), we first reformulate the polynomial predictor into a more concise tensor form:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \sum_{r=2}^{R} \langle \mathcal{W}^r, \otimes_r \boldsymbol{x} \rangle, \tag{4.6}$$

where $\langle \mathcal{W}, \mathcal{V} \rangle$ is the inner product of two same-sized tensors $\mathcal{W}, \mathcal{V} \in \mathbb{R}^{K_1 \times \cdots \times K_r}$, which is defined as the sum of the products of their entries. It is observed that the tensor $\otimes_r \boldsymbol{x}$ comprises all the degree-r monomials in $\boldsymbol{x}$. Therefore, any degree-r homogeneous polynomial predictor satisfies $\langle \mathcal{W}^r, \otimes_r \boldsymbol{x} \rangle$ for some r-order tensor $\mathcal{W}^r$; likewise, any r-order tensor $\mathcal{W}^r$ determines a degree-r homogenous polynomial predictor. This equivalence between polynomials and tensors motivates us to transform the parameter learning of polynomial predictor into tensor learning.

Rather than estimating the variable interactions in tensors independently, an alternative method is tensor decomposition [81] which breaks the independence of interaction parameters and estimates the reliable interaction parameters under high sparsity. Tensor decomposition

is widely used in tensor machines [161] for sparse data based regression, which circumvents the parameter storage issue and achieves better generalization in practice. We then embrace the rank-one tensor decomposition [81] in our next step of tensor learning for consideration of two aspects: the high sparsity of activations in deeper layers of CNNs and the parameter sharing of convolutional filters.

We first briefly review the notations and definitions in the area of rank-one tensor decomposition: the outer product of vectors $\boldsymbol{u}_1 \in \mathbb{R}^{K_1}, \ldots, \boldsymbol{u}_r \in \mathbb{R}^{K_r}$ is the $K_1 \times \cdots \times K_r$ rank-one tensor that satisfies $(\boldsymbol{u}_1 \otimes \cdots \otimes \boldsymbol{u}_r)_{k_1 \ldots, k_r} = (\boldsymbol{u}_1)_{k_1} \cdots (\boldsymbol{u}_r)_{k_r}$. The rank-one decomposition for a tensor $\mathcal{W}$ is defined as $\mathcal{W} = \sum_{d=1}^{D} \alpha^d \boldsymbol{u}_1^d \otimes \cdots \otimes \boldsymbol{u}_r^d$, where α^d is the weight for d-th rank-one tensor, D is the rank of the tensor if D is minimal. We then apply the rank-one approximation [81] for each r-order tensor $\mathcal{W}^r$ and present the following alternative form of polynomial predictor:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \sum_{r=2}^{R} \langle \sum_{d=1}^{D^r} \alpha^{r,d} \boldsymbol{u}_1^{r,d} \otimes \cdots \otimes \boldsymbol{u}_r^{r,d}, \otimes_r \boldsymbol{x} \rangle. \tag{4.7}$$

In order to learn $\boldsymbol{w}$, $\alpha^{r,d}$ and $\boldsymbol{u}_s^{r,d}$ $(r = 2, \ldots, R, s = 1, \ldots, r, d = 1, \ldots, D^r)$, in next section, we show that all the parameters can be absorbed into the conventional trainable modules in CNNs.

4.3.3 Trainable Polynomial Modules

According to the tensor algebra, the Eqn. (4.7) can be further rewritten as:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \sum_{r=2}^{R} \sum_{d=1}^{D^r} \alpha^{r,d} \prod_{s=1}^{r} \langle \boldsymbol{u}_s^{r,d}, \boldsymbol{x} \rangle \tag{4.8}$$

$$= \langle \boldsymbol{w}, \boldsymbol{x} \rangle + \sum_{r=2}^{R} \langle \boldsymbol{\alpha}^r, \boldsymbol{z}^r \rangle \tag{4.9}$$

where the d-th element of the vector $\boldsymbol{z}^r \in \mathbb{R}^{D_r}$ is $\prod_{s=1}^{r} \langle \boldsymbol{u}_s^{r,d}, \boldsymbol{x} \rangle$ which characterizes the degree-r variable interactions under a single rank-one tensor basis. $\boldsymbol{\alpha}^r = [\alpha^{r,1}, \ldots, \alpha^{r,D^r}]^T$ is the associated weight vector of all D^r rank-one tensors. A key observation of Eqns. (4.8), (4.9) is

that we are able to decouple the parameters into $\{\boldsymbol{w}, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^R\}$ and $\{\{\boldsymbol{u}_s^{r,d}\}_{s=1,\ldots,r;d=1,\ldots D_r}\}_{r=2,\ldots,R}$. Notice that for each s, we can first deploy $\{\boldsymbol{u}_s^{r,d}\}_{d=1,\ldots D_r}$ as a set of D_r 1×1 convolutional filters on $\boldsymbol{\mathcal{X}}$ to generate a set of feature maps $\mathcal{Z}_s^r$ of dimension $D^r \times M \times N$. Then, the feature maps $\{\mathcal{Z}_s^r\}_{s=1,\ldots,r}$ from different ss are combined by element-wise product to obtain $\mathcal{Z}^r = \mathcal{Z}_1^r \odot \cdots \odot \mathcal{Z}_r^r$. Therefore, $\{\boldsymbol{u}_s^{r,d}\}_{s=1,\ldots,r;d=1,\ldots D_r}$ can be treated as a polynomial module in learning degree-r polynomial features. As for the former parameter group, it can be easily embedded into the learning of the classifier for the concatenated polynomial features. Refering to Eqn. (4.8), the derivatives for $\boldsymbol{x}$ and each degree-r convolutional filter $\boldsymbol{u}_s^{r,d}$ in back propagation process can be achieved by:

$$\frac{\partial \ell}{\partial \boldsymbol{x}} = \frac{\partial \ell}{\partial \boldsymbol{y}^r} \sum_{d=1}^{D^r} \sum_{s=1}^{r} \Big(\prod_{t \neq s} \langle \boldsymbol{u}_t^{r,d}, \boldsymbol{x} \rangle\Big) \boldsymbol{u}_s^{r,d} \tag{4.10}$$

$$\frac{\partial \ell}{\partial \boldsymbol{u}_s^{r,d}} = \frac{\partial \ell}{\partial \boldsymbol{y}^r} \Big(\prod_{t \neq s} \langle \boldsymbol{u}_t^{r,d}, \boldsymbol{x} \rangle\Big) \boldsymbol{x} \tag{4.11}$$

where $\boldsymbol{y}^r = g(\mathcal{Z}^r) = g(\{z^r\})$ is the pooled feature representation for degree-r polynomial module, ℓ is the loss associated with $\boldsymbol{y}^r$. On this basis, we can embrace those polynomial modules with the trainable CNN architectures and are able to model the higher-order part statistics of any degree. Even though the dominant level of those highly-correlated parts will be enhanced with a larger r, the high-order tensor usually needs large D^r to guarantee a good approximation. Therefore, a relative small degree r should be considered in practice because a high-degree polynomial module increases the computational cost in back propagation, *i.e.*, Eqns. (4.10), (4.11), and the induced high dimensionality of feature would cause over-fitting.

4.4 Hierarchical Convolutional Activations

4.4.1 Higher-order Integration Using Kernel Fusion

The polynomial predictor provides a good measure for the highly-correlated parts but the activations on individual convolutional layer are not sufficient to describe the part relations

from different levels of abstraction and scale. Consequently, we investigate a kernel fusion scheme to combine the hierarchical convolutional activations. Suppose that the local activation descriptor sets from L convolutional layers at spatial correspondences for two images are denoted as $\boldsymbol{\psi}_I : \{\boldsymbol{x}^l\}_{l=1}^L$ and $\boldsymbol{\psi}_{\bar{I}} : \{\bar{\boldsymbol{x}}^l\}_{l=1}^L$. Then we generalize ϕ under linear factorization to fuse the local activations from multiple convolutional layers as below:

$$\begin{aligned} k(\boldsymbol{\psi}_{\boldsymbol{I}}, \boldsymbol{\psi}_{\bar{\boldsymbol{I}}}) &= \langle \phi(\{\boldsymbol{x}^l\}_{l=1}^L), \phi(\{\bar{\boldsymbol{x}}^l\}_{l=1}^L) \rangle \\ &= \sum_{l=1}^{L} \eta_l \langle \phi^l(\boldsymbol{x}^l), \phi^l(\bar{\boldsymbol{x}}^l) \rangle, \end{aligned} \tag{4.12}$$

where η_l is the weight for the matching scores in l-th layer. The above kernel fusion can be re-interpreted as performing polynomial feature extraction at each layer and fusing them in latter phase. Recently, hypercolumn [60] suggests a simple feature concatenation manner to combine different feature maps in CNNs for pixel-level classification, which motivates us to adopt the similar way in our polynomial kernel fusion. Thereby, we assume a holistic mapping ϕ for all layers, *i.e.*, $\sum_{l=1}^L \sqrt{\eta_l}\phi^l(\boldsymbol{x}^l) \rightarrow \phi(\text{concat}(\boldsymbol{x}^1, \ldots, \boldsymbol{x}^L))$ with weights $\sqrt{\eta_l}$s be merged into element-wise scale layers. It should be noted that the spatial resolutions of different convolutional layers need to be consistent for concatenation operation. Alternatively, we can add pooling layers or spatial resampling layers to meet this requirement. In this sense, the expansion of ϕ by Eqn. (4.4) yields two groups of variable interactions: $\prod_{k_l} x_{k_l}^l$ that characterizes the interactions of parts in the l-th layer; and $\prod_{k_l,k_q} x_{k_l}^l x_{k_q}^q$ (where $l \neq q$) that captures additional information of multi-scale part relations from the l-th layer and q-th layer.

4.4.2 Integration Architecture for Deeper Layers

Although the kernel fusion scheme enables polynomial predictor for integrating hierarchical convolutional activations, it may not perform and scale well in the case where large numbers of layers involved. We argue that only the convolutional activations from very deep layers refer to the responses of discriminative semantic parts. That is consistent with the recent studies

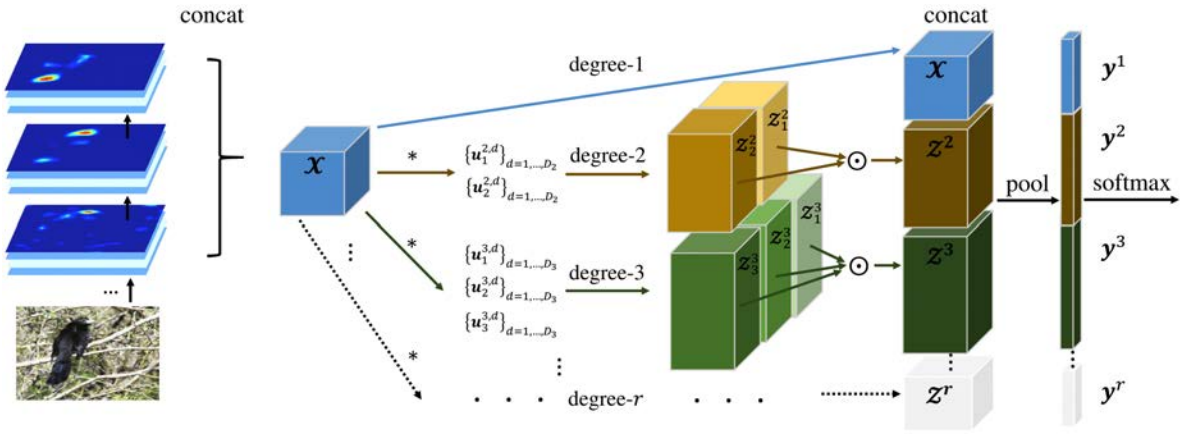


Figure 4.2 Illustration of our integration framework. The convolutional activation maps are concatenated as $\boldsymbol{\mathcal{X}} = \text{concat}(\boldsymbol{\mathcal{X}}^1, \ldots, \boldsymbol{\mathcal{X}}^L)$ and fed into different branches. For r-th branch ($r \geq 2$), the degree-r polynomial module consisting of r groups of 1×1 convolutional filters is deployed to obtain r sets of feature maps $\{\mathcal{Z}_s^r\}_{s=1,\ldots,r}$. Then $\{\mathcal{Z}_s^r\}_{s=1,\ldots,r}$ are integrated as $\mathcal{Z}^r$ by applying element-wise product $\odot$. At last, $\boldsymbol{\mathcal{X}}$ and all $\mathcal{Z}^r$s are concatenated as the degree-r polynomial features, following by sum pooling layer to obtain the pooled representation $\boldsymbol{y} = \text{concat}(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^L)$ with the dimension of $\sum_{r=1}^{R} D_r$ (D_1 denotes the channel number of $\boldsymbol{\mathcal{X}}$), and softmax layer.

[144, 201] which regard the convolutional filters in deeper layers as weak part detectors. In our experiments, we demonstrate that the integration of the last three convolutional activation layers (*i.e.*, *relu5_1*, *relu5_2*, and *relu5_3* in VGG-16 model [146]) is fairly effective to obtain satisfactory performance. Even though lower layers could be involved, the effect is less obvious on the improvement but higher computational complexity in both the training and testing phases. Fig. 4.2 presents our CNN architecture for integrating multiple convolutional layers. Compared with the B-CNN methods [42, 94] focusing only on the degree-2 part statistics, our approach provides a general solution to model complex part interactions from hierarchical layers in different degrees and its superiority will be demonstrated in experiments.

4.5 Experimental Results

In this section, we evaluate the effectiveness of our proposed integration framework on three fine-grained categorization datasets: Caltech-UCSD Bird-200-2011 (CUB) [167], Aircraft [104] and Cars [83]. The experimental comparisons with state-of-the-art methods indicate that effective feature integration from CNN is a promising solution for FGVC in contrast with the requirements of massive external data or detailed part annotation.

4.5.1 Datasets and Implementation Details

CUB dataset contains 11,788 bird images. There are altogether 200 bird species, and the number of images per class is about 60. The significant variations in pose, viewpoint and illumination inside each class make this dataset very challenging. We adopt the publicly available split [167], which use nearly half of the dataset for training and the other half for testing.

Aircraft dataset has 100 different aircraft model variants, giving 100 images for each model. The aircrafts appear at different scales, design structures and appearances. We adopt the training/testing split protocol provided by [104] to perform our experiments.

Cars dataset consists of 16,185 images from 196 car classes. Each class has about 80 images with different car sizes and heavy clutter background. We use the same split provided by [83], divided with 8,144 images for training and 8,041 images for testing.

Implementation details: our networks on all datasets are fine-tuned on the VGG-16 model pre-trained on ILSVRC-2012 dataset [135] for a fair comparison with most state-of-the-art FGVC methods. The framework can also be applied to the recently proposed network architectures such as Inception [158] and ResNet [61]. We remove the last three fully-connected layers and construct a directed acyclic graph (DAG) to combine all the components in our framework. Before fed into softmax layer, we first pass pooled polynomial features through

ℓ_2 normalization step. We then use logistic regression to initialize the parameters of classification layer, and adopt Rademacher vectors (*i.e.*, each of its components is chosen independently using a fair coin toss from the set $\{-1, 1\}$) as proper initializations [75] of homogenous polynomial kernels for the 1×1 convolutional filters. In the training phase, following [94], we transform the input image by cropping the largest image region around its center, resizing it to 448×448, and creating its mirrored version to double the training set. During fine-tuning, the learning rates of those pre-trained VGG-16 layers and the newly added layers, including 1×1 convolutional layers and classification layer, are initialized as 0.001. We train all the networks using stochastic gradient descent with a batch size of 16, momentum of 0.9. In the testing phase, we follow the popular CNN-SVM scheme [94], *i.e.*, use softmax loss in training and then perform the evaluation on the extracted features by SVM. Our code is implemented on the open source MatConvNet framework with a single NVIDIA GeForce GTX TITAN X GPU and can be downloaded at `http://www4.comp.polyu.edu.hk/~cslzhang/code/hihca.zip`.

4.5.2 Analysis of the Proposed Framework

Effect of number of 1×1 convolutional filters. To validate the effectiveness of introducing tensor decomposition in our polynomial predictor, we investigate the effect of different D^r for the approximation of each r-order tensor $\mathcal{W}^r$. We first evaluate the classification accuracies on the CUB dataset on a single layer *relu5_3* using different homogeneous polynomial kernels for solely modeling the degree-r variable interactions, *i.e.*, x_i, $x_i x_j$, $x_i x_j x_k$, $x_i x_j x_k x_l$. The number D^r for degree-r convolutional filters varies from 512 to 32,768. The results are shown in Fig. 4.3. As expected, increasing D^r leads higher accuracies on all degrees. Interestingly, when D^r is small, degree-2 always leads a higher accuracy than those with higher degrees, which indicates that modelling higher-order part interactions often yields a tensor of dense parameters. It is observed that the performance gain is slight when the number D^r increases from 8,192 to 32,768, which infers that a relative sparse tensor $\mathcal{W}^r$ can comprehensively encode the

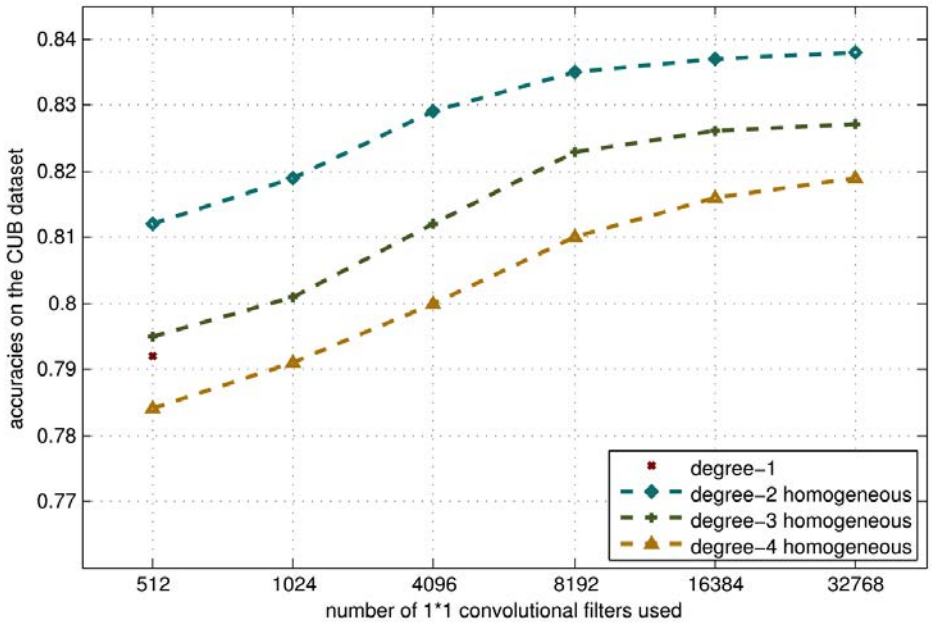


Figure 4.3 Accuracies achieved by using polynomial kernels with varied numbers of 1×1 convolutional filters on the CUB dataset.

distinguishing part interactions of fine-grained objects from the very sparse activation features. Therefore, we uniformly use 8,192 1×1 convolutional filters in all the polynomial modules in consideration of feature dimension, computational complexity as well as accuracy.

Effect of polynomial degree *r*. We further demonstrate the superiority of using higher-order part interactions both with and without finetuning on the CUB dataset in Table 4.1. We observe that the degree-2 polynomial kernel significantly outperforms the linear kernel. It implies that the co-occurrence statistics is very effective in capturing part relations, which is more informative in distinguishing objects with homogeneous appearance than the simple part occurrence statistics. The accuracy degrades considerably as the degree r increases from 2 to 6, which might be explained by the fact the low-degree interactions with high counts are more reliable. As the reliable high-degree interactions are usually a few in number, the sum pooling will abate those scarce interactions in the pooled polynomial representation, which weakens the discriminative ability of the final concatenated representation. Table. 4.2 lists

the frame-per-second (FPS) comparison in both the training and testing phases using different polynomial kernels. Since there is high computational overhead involved in the polynomial modules in the network, a large degree r will significantly slow the speed. Therefore, we suggest to adopt 2 as the reasonable degree in all the experiments in Section 4.5.3 even though degree-3 kernel can achieve slightly better results on Aircraft and Cars datasets.

Table 4.1 Accuracy comparison with different non-homogeneous polynominal kernels.

r	1	2	3	4	5	6
non-ft	75.7	**78.3**	76.4	74.6	72.4	71.2
ft	79.2	**83.7**	83.3	82.0	81.1	79.5

Table 4.2 FPS with different non-homogeneous polynomial kernels.

r	2	3	4	5	6
Training	9.7	7.4	5.5	4.2	2.8
Testing	29.8	23.7	18.3	14.5	10.4

Effect of feature integration. We then provide details of the results by using higher-order integration for hierarchical convolutional activations. We focus on *relu5_1*, *relu5_2* and *relu5_3* as they exhibit good capacity in capturing semantic part information compared with lower layers. And we analyze the impact factors of layers, kernels, and finetuning on the CUB dataset. The accuracies are obtained under five polynomial kernels including linear kernel, degree-2 homogeneous kernel, degree-2 non-homogeneous (single + pairwise interactions), degree-3 homogeneous kernel and degree-3 non-homogeneous kernel (single + pairwise + triple interactions). We consider the following baselines: *relu5_3* uses only *relu5_3* activations. *relu5_3+relu5_2*, *relu5_3+relu5_1* and *relu5_2+relu5_1* are integration baselines that use 2 layers. *relu5_1+relu5_2+relu5_3* is the full integration of three layers. The results in Table 4.3 demonstrate that the performance gain of our framework comes from three factors: (i) higher-order integration, (ii) finetuning, (iii) multiple layers. Notably, we observe the remarkable

performance benefits on the baseline *relu5_3+relu5_2* and the full model of three layers by exploiting the degree-2 and degree-3 polynomial kernels, which implies that the discriminative power can be enhanced by the complementary capacities of hierarchical convolutional layers compared with the isolated *relu5_3* layer. As the baseline *relu5_3+relu5_2* already presents the best performance, thus we set the feature integration as *relu5_3+relu5_2* in all the experiments in Section 4.5.3.

Table 4.3 Accuracy comparison with different baselines.

	r5_3	*r5_3+ r5_2*	*r5_3+ r5_1*	*r5_2+ r5_1*	*r5_3+ r5_2+ r5_1*
degree-1					
non-ft	75.7	**77.2**	75.5	68.9	77.0
ft	79.2	80.4	79.3	71.1	**80.8**
degree-2 homogeneous					
non-ft	77.2	78.1	77.5	72.3	**78.4**
ft	83.5	**85.0**	83.3	76.0	84.9
degree-2 non-homogeneous					
non-ft	78.3	78.5	77.5	72.1	**78.6**
ft	83.7	**85.3**	83.6	76.5	85.1
degree-3 homogeneous					
non-ft	75.7	**76.9**	76.0	70.7	76.1
ft	82.3	**83.8**	81.5	74.1	83.3
degree-3 non-homogeneous					
non-ft	76.4	**78.2**	77.4	72.3	78.1
ft	83.3	**84.6**	82.1	75.4	84.5

We also compare our higher-order integration with hypercolumn [60] and HED [179] based feature integrations. Since the original hypercolumn and HED are introduced for pixel-wise classification, for fair comparison, we revise hypercolumn as the feature concatenation of *relu5_3*, *relu5_2* and *relu5_1*, following by max pooling (denoted as Hypercolumn*); and revise HED by training classifiers for the pooled activation features at each layer and then fuse the predictions (denoted as HED*). Table 4.4 shows that our integration framework is significantly superior to Hypercolumn* and HED*. This is not surprising since Hypercolumn* and HED* can be treated as degree-1 integration to some extent.

Table 4.4 Accuracy comparison with different feature integrations.

Degree-2 integration	Hypercolumn*	HED*
85.1	80.9	82.3

4.5.3 Comparison with State-of-the-art Methods

Results on the CUB dataset. We first compare our framework along with both the annotation-based methods (*i.e.*, using object bounding boxes or part annotations) and annotation-free methods (*i.e.*, only using image-level labels) on the CUB dataset. As shown in Table 4.5, unlike the state-of-the-art result obtained from SPDA-CNN (85.1%) [193] which relies on the additional annotations of seven parts, we can still achieve a comparable accuracy of 85.3% with only image-level labels and significant improvements over PB R-CNN [199] and FG-Without [82]. Furthermore, our method is slightly inferior to BoostCNN [111] and outperforms all other annotation-free methods with a modest improvement (about 1%) compared with STN [68], B-CNN [94] and PDFS [201]. However, STN [68] uses a better baseline CNN (Inception [158]) than our VGG-16 network and PDFS [201] cannot be trained by end-to-end manner. B-CNN [94] attempts to achieve the feature complementary based on the outer product of convolutional activations from two networks (*i.e.*, VGG-M and VGG-16). However,

Table 4.5 Accuracies (%) on the CUB dataset. "bbox" and "parts" refer to object bounding box and part annotations.

methods	train anno.	test anno.	acc.
PB R-CNN [199]	bbox+parts	n/a	73.9
FG-Without [82]	bbox	bbox	82.0
SPDA-CNN [193]	bbox+parts	bbox+parts	85.1
STN [68]	n/a	n/a	84.1
B-CNN [94]	n/a	n/a	84.1
PDFS [201]	n/a	n/a	84.5
BoostCNN [111]	n/a	n/a	**85.6**
Ours	n/a	n/a	85.3

Table 4.6 Accuracies (%) on the Aircraft and Cars datasets.

methods	acc. (Aircraft)	acc. (Cars)
Symbiotic [18]	72.5	78.0
FV-FGC [49]	80.7	82.7
B-CNN [94]	84.1	91.3 (90.6)
Ours	**88.3**	**91.7**

our framework shows that the better complementarity can be achieved by exploiting the natural hierarchical structures of CNNs. BoostCNN uses BCNN as the base CNN and adopts an ensemble learning method to incorporate boosting weights. Thus, a fair comparison is to use ours as the base CNN in BoostCNN.

Results on the Aircraft and Cars datasets. The methods for the Aircraft and Cars datasets are all annotation-free since there are no ground-truth part annotations on these two datasets. We first evaluate our framework on the Aircraft dataset, and the related results are shown in the second column of Table 4.6. Our network achieves significantly better classifi-

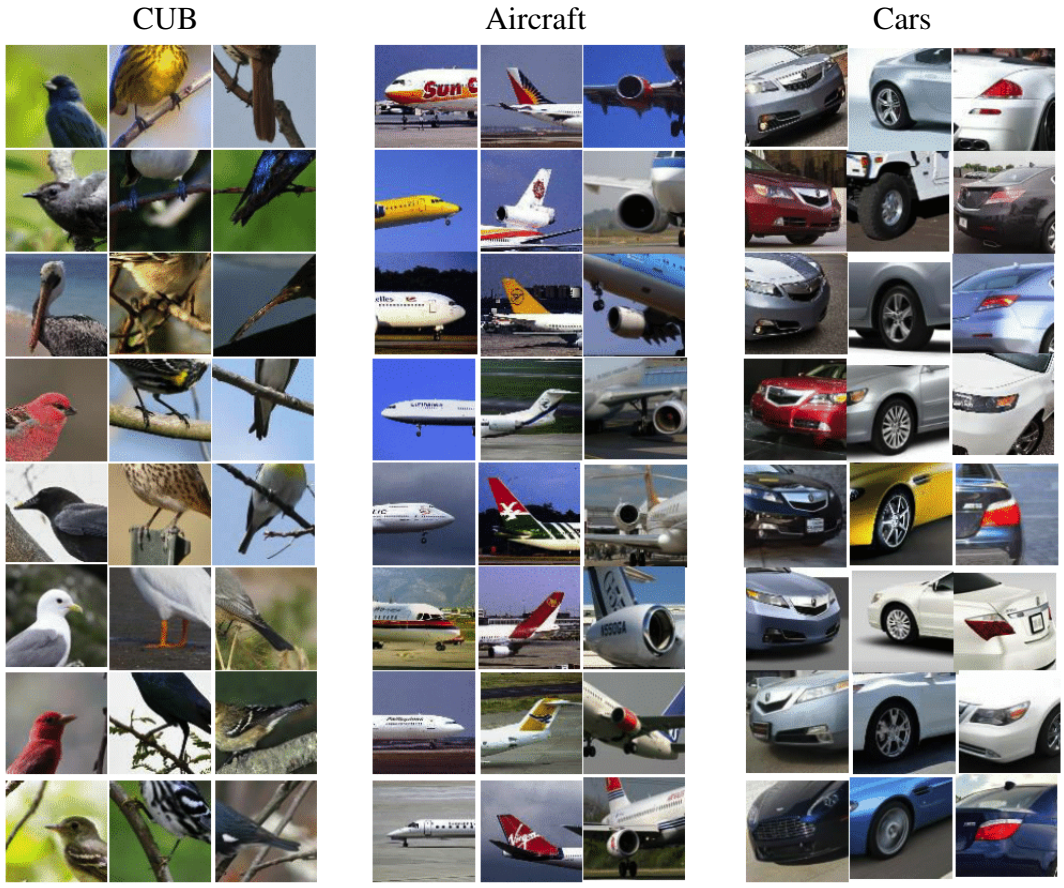


Figure 4.4 Visualization of the learned image patches in our fine-tuned networks on the CUB, Aircraft and Cars datasets.

cation accuracy than the state-of-the-art B-CNN which can be seemed like a specific degree-2 case in our framework. As we find that *relu5_2* instead of *relu5_3* achieves the best performance in Aircraft dataset, our improvement might be due to the reasons: (1) B-CNN only focuses on *relu5_3* where the discriminative parts are highly out-numbered, thus these parts might be overwhelmed by large non-discriminative region in pooling stage; (2) the discriminative parts in this dataset may occur simultaneously in both the coarse and fine scales. Therefore, the rich representation by incorporating multiple layers in our integration framework mitigates the local ambiguities of single-layer representation to a large extent.

The third column of Table 4.6 provides the comparision on the Cars dataset. B-CNN [94]

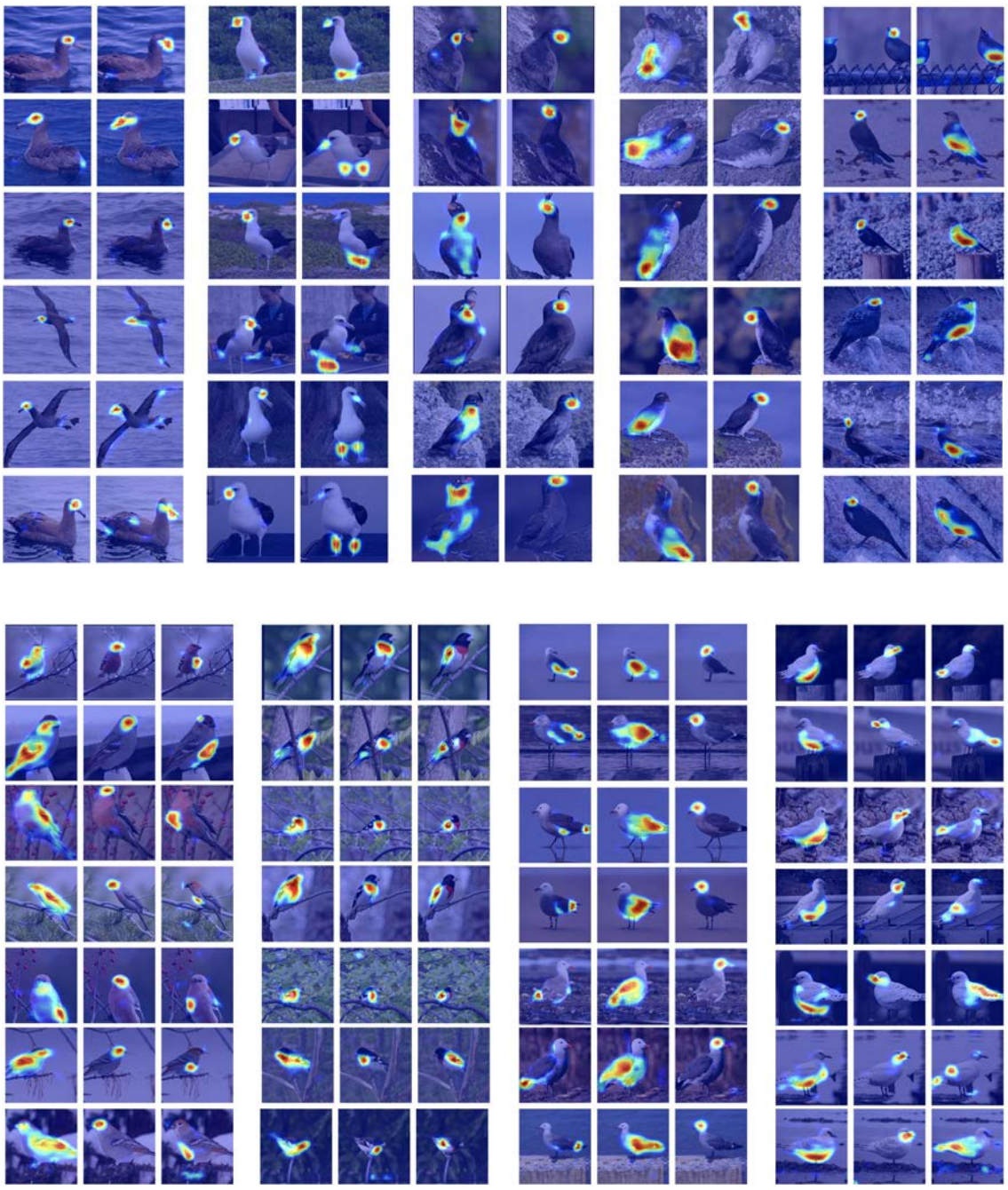


Figure 4.5 The degree-2 and degree-3 part interactions on the CUB dataset.

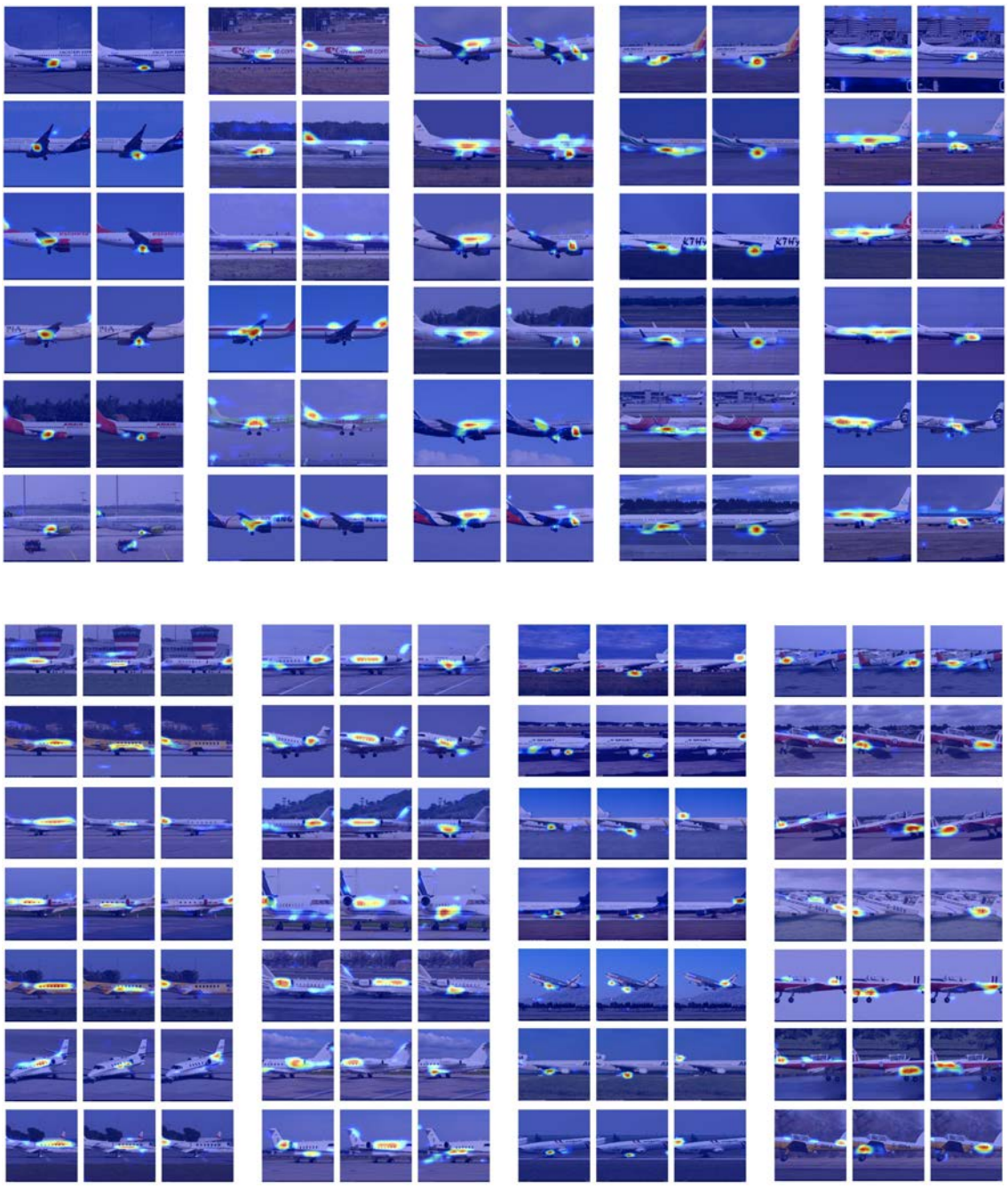


Figure 4.6 The degree-2 and degree-3 part interactions on the Aircraft dataset.

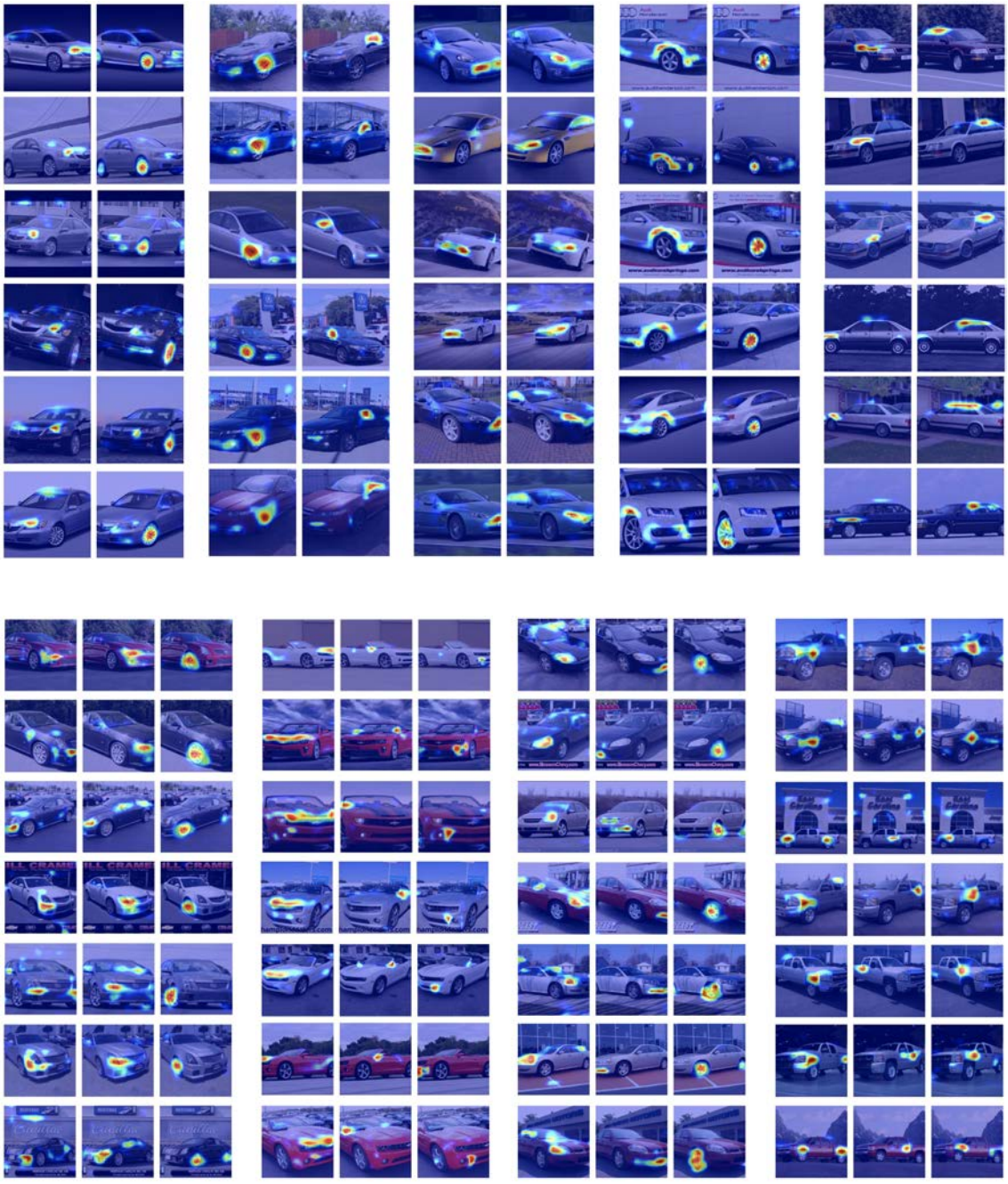


Figure 4.7 The degree-2 and degree-3 part interactions on the Cars dataset.

shows the similar accuracy behavior with ours and both present a large margin over Symbiotic [18] and FV-FGC [49]. The accuracy of B-CNN [94] using two networks is very close to

ours (91.3% vs. 91.7%), yet for the single network case, it still has the accuracy gap of 1.1%, which infers that the hierarchical feature integration on a single network can also contribute the feature complementary as done by two different networks.

Visualization for the learned image patches and part interactions. In Fig. 4.4, we visualize some image patches with the highest activations in the deeper layers of our fine-tuned networks, and the patches in each column come from different feature channels/maps. We obviously observe strong semantic-related parts such as heads, legs and tails in CUB; cockpit, tail stabilizers and engine in Aircraft; front bumpers, wheels and lights in Cars. Furthermore, in Figs. 4.5, 4.6 and 4.7, we also visualize the degree-2 and degree-3 part interactions based on the largest values in classifier parameters in CUB, Aircraft and Cars, respectively. The strong part connections exactly reflect the nature of our approach which aims to improve the feature discrimination by the effective combinations of these parts.

4.6 Conclusion

It is preferred to perform FGVC under a more realistic setting without part annotations and any prior knowledge for explicit object appearance modelling. In this work, by considering the weak parts in CNN itself, we present a novel higher-order integration framework of hierarchical convolutional layers to derive a rich representation for FGVC. Based on the kernel mapping scheme, we propose a polynomial predictor to exploit the higher-order part relations and presented the trainable polynomial modules which can be plugged in conventional CNNs. Furthermore, the higher-order integration framework can be naturally extended to mine the multi-scale part relations in hierarchical layers. The results on the CUB, Aircraft and Cars datasets manifest competitive performance and demonstrate the effectiveness of our integration framework.

Chapter 5

Weakly-supervised Video Summarization using Variational Encoder-Decoder and Web Prior

5.1 Introduction

Recently, it has been attracting much interest in extracting the representative visual elements from a video for sharing on social media, which aims to effectively express the semantics of the original lengthy video. However, this task, often referred to as video summarization, is laborious, subjective and challenging since videos usually exhibit very complex semantic structures, including diverse scenes, objects, actions and their complex interactions.

A noticeable trend appeared in recent years is to use the deep neural networks (DNNs) [57, 197] for video summarization since DNNs have made significant progress in various video understanding tasks [30, 76, 117]. However, annotations used in the video summarization task are in the form of frame-wise labels or importance scores, collecting a large number of annotated videos demands tremendous effort and cost. Consequently, the widely-used

benchmark datasets [23, 151] only cover dozens of well-annotated videos, which becomes a prominent stumbling block that hinders the further improvement of DNNs based summarization techniques. Meanwhile, annotations for summarization task are subjective and not consistent across different annotators, potentially leading to overfitting and biased models. Therefore, the advanced studies toward taking advantage of augmented data sources such as web images [79], GIFs [57] and texts [126], which are complimentary for the summarization purpose.

To drive the techniques along with this direction, we consider an efficient weakly-supervised setting of learning summarization models from a vast number of web videos. Compared with other types of auxiliary source domain data for video summarization, the temporal dynamics in these user-edited "templates" offer rich information to locate the diverse but semantic-consistent visual contents which can be used to alleviate the ambiguities in small-size summarization. These short-form videos are readily available from web repositories (*e.g.*, YouTube) and can be easily collected using a set of topic labels as search keywords. Additionally, these web videos have been edited by a large community of users, the risk of building a biased summarization model is significantly reduced. Several existing works [23, 120] have explored different strategies to exploit the semantic relatedness between web videos and benchmark videos. So motivated, we aim to effectively utilize the large collection of weakly-labelled web videos in learning more accurate and informative video representations which: (i) preserve essential information within the raw videos; (ii) contain discriminative information regarding the semantic consistency with web videos. Therefore, the desired deep generative models are necessitated to capture the underlying latent variables and make practical use of web data and benchmark data to learn abstract and high-level representations.

To this end, we present a generative framework for summarizing videos in this paper, which is illustrated in Fig. 5.1. The basic architecture consists of two components: a variational autoencoder (VAE) [80] model for learning the latent semantics from web videos; and

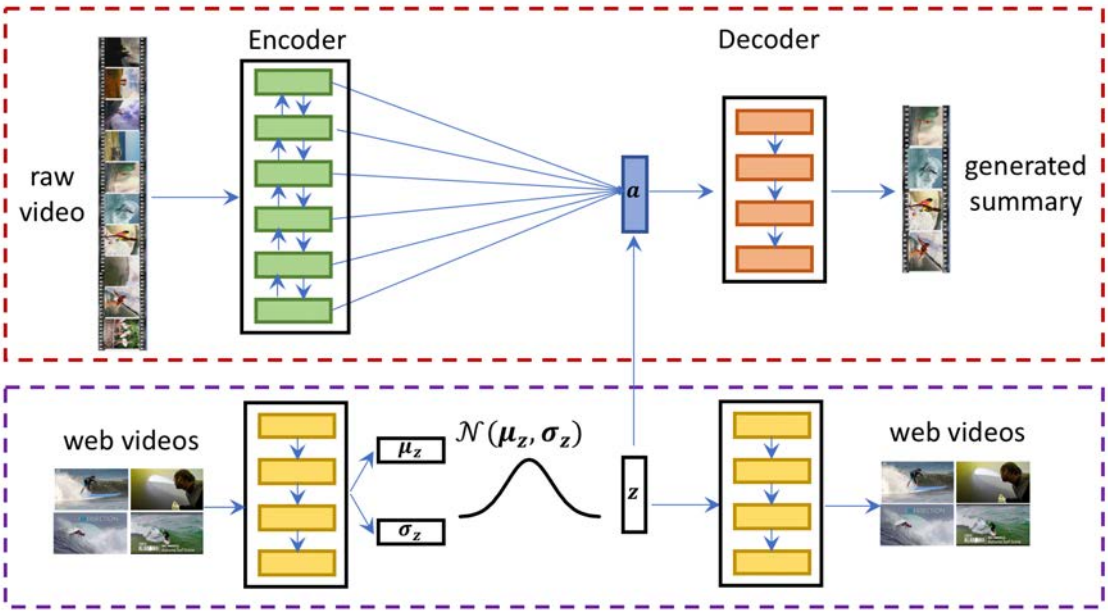


Figure 5.1 An illustration of the proposed generative framework for video summarization. A VAE model is pre-trained on web videos (purple dashed rectangle area); And the summarization is implemented within an encoder-decoder paradigm by using both the attention vector and the sampled latent variable from VAE (red dashed rectangle area).

a sequence encoder-decoder with attention mechanism for summarization. The role of VAE is to map the videos into a continuous latent variable, via an inference network (encoder), and then use the generative network (decoder) to reconstruct the input videos conditioned on samples from the latent variable. For the summarization component, the association is temporally ambiguous since only a subset of fragments in the raw video is relevant to its summary semantics. To filter out the irrelevant fragments and identify informative temporal regions for the better summary generation, we exploit the soft attention mechanism where the attention vectors (*i.e.*, context representations) of raw videos are obtained by integrating the latent semantics trained from web videos. Furthermore, we provide a weakly-supervised semantic matching loss instead of reconstruction loss to learn the topic-associated summaries in our generative framework. In this sense, we take advantage of potentially accurate and flexible

latent variable distribution from external data thus strengthen the expressiveness of generated summary in the encoder-decoder based summarization model. To evaluate the effectiveness of the proposed method, we comprehensively conduct experiments using different training settings and demonstrate that our method with web videos achieves significantly better performance than competitive video summarization approaches.

5.2 Related Work

5.2.1 Video Summarization

Video summarization is a challenging task which has been explored for many years [112, 164] and can be grouped into two broad categories: unsupervised and supervised learning methods. Unsupervised summarization methods focus on low-level visual cues to locate the important segments of a video. Various strategies have been investigated, including clustering [54, 55], sparse optimizations [33, 121], and energy minimization [37, 129]. A majority of recent works mainly study the summarization solutions based on the supervised learning from human annotations. For instance, to make a large-margin structured prediction, submodular functions are trained with human-annotated summaries [56]. Gygli *et al.* [55] propose a linear regression model to estimate the interestingness score of shots. Gong *et al.* [46] and Sharghi *et al.* [142] learn from user-created summaries for selecting informative video subsets. Zhang *et al.* [196] show summary structures can be transferred between videos that are semantically consistent. More recently, DNNs based methods have been applied for video summarization with the help of pairwise deep ranking model [190] or recurrent neural networks (RNNs) [197]. However, these approaches assume the availability of a large number of human-created video-summary pairs or fine-grained temporal annotations, which are in practice difficult and expensive to acquire. Alternatively, there have been attempts to leverage information from other data sources such as web images, GIFs and texts [57, 79, 126]. Chu *et al.* [23] propose

to summarize shots that co-occur among multiple videos of the same topic. Panda *et al.* [119] present an end-to-end 3D convolutional neural network (CNN) architecture to learn summarization model with web videos. In this paper, we also consider to use the topic-specific cues in web videos for better summarization, but adopt a generative summarization framework to exploit the complementary benefits in web videos.

5.2.2 Video Highlight Detection

Video highlight detection is highly related to video summarization and many earlier approaches have primarily been focused on specific data scenarios such as broadcast sport videos [134, 160]. Traditional methods usually adopt the mid-level and high-level audio-visual features due to the well-defined structures. For general highlight detection, Sun *et al.* [156] employ a latent SVM model detect highlights by learning from pairs of raw and edited videos. The DNNs also have achieved big performance improvement and shown great promise in highlight detection [184]. However, most of these methods treat highlight detection as a binary classification problem, while highlight labelling is usually ambiguous for humans. This also imposes heavy burden for humans to collect a huge amount of labelled data for training DNN based models.

5.2.3 Deep Generative Models

Deep generative models are very powerful in learning complex data distribution and low-dimensional latent representations. Besides, the generative modelling for video summarization might provide an effective way to bring scalability and stability in training a large amount of web data. Two of the most effective approaches are VAE [80] and generative adversarial network (GAN) [48]. VAE aims at maximizing the variational lower bound of the observation while encouraging the variational posterior distribution of the latent variables to be close to the prior distribution. A GAN is composed of a generative model and a discriminative

model and trained in a min-max game framework. Both VAE and GAN have already shown promising results in image/frame generation tasks [106, 133, 166]. To embrace the temporal structures into generative modelling, we propose a new variational sequence-to-sequence encoder-decoder framework for video summarization by capturing both the video-level topics and web semantic prior. The attention mechanism embedded in our framework can be naturally used as key shots selection for summarization. Most related to our generative summarization is the work of Mahasseni *et al.* [100], who present an unsupervised summarization in the framework of GAN. However, the attention mechanism in their approach depends solely on the raw video itself thus has the limitation in delivering diverse contents in video-summary reconstruction.

5.3 VESD Model

As an intermediate step to leverage abundant user-edited videos on the Web to assist the training of our generative video summarization framework, in this section, we first introduce the basic building blocks of the proposed framework, called variational encoder-summarizer-decoder (VESD). The VESD consists of three components: (i) an encoder RNN for raw video; (ii) an attention-based summarizer for raw video; (iii) a decoder RNN for summary video.

Following the video summarization pipelines in previous methods [127, 197], we first perform temporal segmentation and shot-level feature extraction for raw videos using CNNs. Each video $\mathcal{X}$ is then treated as a sequential set of multiple non-uniform shots, where $\boldsymbol{x}_t$ is the feature vector of the t-th shot in video representation $\boldsymbol{X}$. Most supervised summarization approaches aim to predict labels/scores which indicate whether the shots should be included in the summary, however, suffering from the drawbacks of selection of redundant visual contents. For this reason, we formulate video summarization as video generation task which allows the summary representation $\boldsymbol{Y}$ does not necessarily be restricted to a subset of $\boldsymbol{X}$. In this manner, our method centres on the semantic essence of a video and can exhibit the high

tolerance for summaries with visual differences. Following the encoder-decoder paradigm [157], our summarization framework is composed of two parts: the encoder-summarizer is an inference network $q_{\phi}(\boldsymbol{a}|\boldsymbol{X}, \boldsymbol{z})$ that takes both the video representation $\boldsymbol{X}$ and the latent variable $\boldsymbol{z}$ (sampled from the VAE module pre-trained on web videos) as inputs. Moreover, the encoder-summarizer is supposed to generate the video content representation $\boldsymbol{a}$ that captures all the information about $\boldsymbol{Y}$. The summarizer-decoder is a generative network $p_{\theta}(\boldsymbol{Y}|\boldsymbol{a}, \boldsymbol{z})$ that outputs the summary representation $\boldsymbol{Y}$ based on the attention vector $\boldsymbol{a}$ and the latent representation $\boldsymbol{z}$.

5.3.1 Encoder-Summarizer

To date, modelling sequence data with RNNs has been proven successful in video summarization [197]. Therefore, for the encoder-summarizer component, we employ a pointer RNN, *e.g.*, a bidirectional Long Short-Term Memory (LSTM), as an encoder that processes the raw videos, and a summarizer aims to select the shots of most probably containing salient information. The summarizer is exactly the attention-based model that generates the video context representation by attending to the encoded video features.

In time step t, we denote $\boldsymbol{x}_t$ as the feature vector for the t-th shot and $\boldsymbol{h}_t^e$ as the state output of the encoder. It is known that $\boldsymbol{h}_t^e$ is obtained by concatenating the hidden states from each direction:

$$\boldsymbol{h}_t^e = [\mathrm{RNN}_{\overrightarrow{enc}}(\overrightarrow{\boldsymbol{h}_{t-1}}, \boldsymbol{x}_t); \mathrm{RNN}_{\overleftarrow{enc}}(\overleftarrow{\boldsymbol{h}_{t+1}}, \boldsymbol{x}_t)]. \tag{5.1}$$

The attention mechanism is proposed to compute an attention vector $\boldsymbol{a}$ of input sequence by summing the sequence information $\{\boldsymbol{h}_t^e, t = 1, \ldots, |\boldsymbol{X}|\}$ with the location variable $\boldsymbol{\alpha}$ as follows:

$$\boldsymbol{a} = \sum_{t=1}^{|\boldsymbol{X}|} \alpha_t \boldsymbol{h}_t^e, \tag{5.2}$$

where α_t denotes the t-th value of $\boldsymbol{\alpha}$ and indicates whether the t-th shot is included in summary or not. As mentioned in [181], when using the generative modelling on the log-likelihood of the conditional distribution $p(\boldsymbol{Y}|\boldsymbol{X})$, one approach is to sample attention vector $\boldsymbol{a}$ by assigning

the Bernoulli distribution to $\boldsymbol{\alpha}$. However, the resultant Monte Carlo gradient estimator of the variational lower-bound objective requires complicated variance reduction techniques and may lead to unstable training. Instead, we adopt a deterministic approximation to obtain $\boldsymbol{a}$. That is, we produce an attentive probability distribution based on $\boldsymbol{X}$ and $\boldsymbol{z}$, which is defined as $\alpha_t := p(\alpha_t|\boldsymbol{h}_t^e, \boldsymbol{z}) = \text{softmax}(\varphi_t([\boldsymbol{h}_t^e; \boldsymbol{z}]))$, where $\boldsymbol{\varphi}$ is a parameterized potential typically based on a neural network, *e.g.*, multilayer perceptron (MLP). Accordingly, the attention vector in Eqn. (5.2) turns to:

$$\boldsymbol{a} = \sum_{t=1}^{N} p(\alpha_t|\boldsymbol{h}_t^e, \boldsymbol{z})\boldsymbol{h}_t^e, \tag{5.3}$$

which is fed to the decoder RNN for summary generation. The attention mechanism extracts an attention vector $\boldsymbol{a}$ by iteratively attending to the raw video features based on the latent variable $\boldsymbol{z}$ learned from web data. In doing so the model is able to adapt to the ambiguity inherent in summaries and obtain salient information of raw video through attention. Intuitively, the attention scores α_ts are used to perform shot selection for summarization.

5.3.2 Summarizer-Decoder

We specify the summary generation process as $p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, \boldsymbol{z})$ which is the conditional likelihood of the summary given the attention vector $\boldsymbol{a}$ and the latent variable $\boldsymbol{z}$. Different with the standard Gaussian prior distribution adopted in VAE, $p(\boldsymbol{z})$ in our framework is pre-trained on web videos to regularize the latent semantic representations of summaries. Therefore, the summaries generated via $p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, \boldsymbol{z})$ are likely to possess diverse contents. In this manner, $p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, \boldsymbol{z})$ is then reconstructed via a RNN decoder at each time step t: $p_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{a}, [\boldsymbol{\mu}_z, \boldsymbol{\sigma}_z^2])$, where $\boldsymbol{\mu}_z$ and $\boldsymbol{\sigma}_z$ are nonlinear functions of the latent variables specified by two learnable neural networks (detailed in Section 5.4).

5.3.3 Variational Inference

Given the proposed VESD model, the network parameters $\{\boldsymbol{\phi}, \boldsymbol{\theta}\}$ need to be updated during inference. We marginalize over the latent variables $\boldsymbol{a}$ and z by maximizing the following variational lower-bound $\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta})$

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{a}, z|\boldsymbol{X}, \boldsymbol{Y})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, z) - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{a}, z|\boldsymbol{X}, \boldsymbol{Y})|p(\boldsymbol{a}, z))], \tag{5.4}$$

where $\mathrm{KL}(\cdot)$ is the Kullback-Leibler divergence. We assume the joint distribution of the latent variables $\boldsymbol{a}$ and z has a factorized form, *i.e.*, $q_{\boldsymbol{\phi}}(\boldsymbol{a}, z|\boldsymbol{X}, \boldsymbol{Y}) = q_{\boldsymbol{\phi}^{(z)}}(z|\boldsymbol{X}, \boldsymbol{Y}) q_{\boldsymbol{\phi}^{(a)}}(\boldsymbol{a}|\boldsymbol{X}, \boldsymbol{Y})$, and notice that $p(\boldsymbol{a}) = q_{\boldsymbol{\phi}^{(a)}}(\boldsymbol{a}|\boldsymbol{X}, \boldsymbol{Y})$ is defined with a deterministic manner in Section 5.3.1. Therefore the variational objective in Eqn. (5.4) can be derived as:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) &= \mathbb{E}_{q_{\boldsymbol{\phi}^{(z)}}(z|\boldsymbol{X}, \boldsymbol{Y})}[\mathbb{E}_{q_{\boldsymbol{\phi}^{(a)}}(\boldsymbol{a}|\boldsymbol{X}, \boldsymbol{Y})} \log p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, z) \\
&\quad -\mathrm{KL}(q_{\boldsymbol{\phi}^{(a)}}(\boldsymbol{a}|\boldsymbol{X}, \boldsymbol{Y})||p(\boldsymbol{a}))] + \mathrm{KL}(q_{\boldsymbol{\phi}^{(z)}}(z|\boldsymbol{X}, \boldsymbol{Y})||p(z)) \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(z|\boldsymbol{X}, \boldsymbol{Y})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, z)] + \mathrm{KL}(q_{\boldsymbol{\phi}}(z|\boldsymbol{X}, \boldsymbol{Y})||p(z)).
\end{aligned} \tag{5.5}$$

The above variational lower-bound offers a new perspective for exploiting the reciprocal nature of raw video and its summary. Maximizing Eqn. (5.5) strikes a balance between minimizing generation error and minimizing the KL divergence between the approximated posterior $q_{\boldsymbol{\phi}^{(z)}}(z|\boldsymbol{X}, \boldsymbol{Y})$ and the prior $p(z)$.

5.4 Weakly-supervised VESD

In practice, as only a few video-summary pairs are available, the latent variable z cannot characterize the inherent semantic in video and summary accurately. Motivated by the VAE/GAN model [86], we explore a weakly-supervised learning framework and endow our VESD the ability to make use of rich web videos for the latent semantic inference. The VAE/GAN model extends VAE with the discriminator network in GAN, which provides a method that

constructs the latent space from inference network of data rather than random noises and implicitly learns a rich similarity metric for data. The similar idea has also been investigated in [100] for unsupervised video summarization. Recall that the discriminator in GAN tries to distinguish the generated examples from real examples; Following the same spirit, we apply the discriminator in the proposed VESD which naturally results in minimizing the following adversarial loss function:

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\psi}) \quad = \quad -\mathbb{E}_{\hat{\boldsymbol{Y}}}[\log \mathrm{D}_{\boldsymbol{\psi}}(\hat{\boldsymbol{Y}})] - \mathbb{E}_{\boldsymbol{X},z}[\log(1 - \mathrm{D}_{\boldsymbol{\psi}}(\boldsymbol{Y}))], \tag{5.6}$$

where $\hat{\boldsymbol{Y}}$ refers to the representation of web video. Unfortunately, the above loss function suffers from the unstable training in standard GAN models and cannot be directly extended into supervised scenario. To address these problems, we propose to employ a semantic feature matching loss for the weakly-supervised setting of VESD framework. The objective requires the representation of generated summary to match the representation of web videos under a similarity function. For the prediction of the semantic similarity, we replace $p_{\boldsymbol{\theta}}(\boldsymbol{Y}|\boldsymbol{a}, z)$ with the following sigmoid function:

$$p_{\boldsymbol{\theta}}(c|\boldsymbol{a}, \boldsymbol{h}^d(\hat{\boldsymbol{Y}})) = \sigma(\boldsymbol{a}^T \boldsymbol{M} \boldsymbol{h}^d(\hat{\boldsymbol{Y}})), \tag{5.7}$$

where $\boldsymbol{h}^d(\hat{\boldsymbol{Y}})$ is the last output state of $\hat{\boldsymbol{Y}}$ in the decoder RNN and $\boldsymbol{M}$ is the sigmoid parameter. We randomly pick $\hat{\boldsymbol{Y}}$ in web videos and c is the pair relatedness label, *i.e.*, $c = 1$ if $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$ are semantically matched. We can also generalize the above matching loss to multi-label case by replacing c with one-hot vector $\boldsymbol{c}$ whose nonzero position corresponds the matched label. Therefore, the objective (5.5) can be rewritten as:

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\psi}) \quad = \quad \mathbb{E}_{q_{\boldsymbol{\phi}}(z)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{c}|\boldsymbol{a}, \boldsymbol{h}^d(\hat{\boldsymbol{Y}}))] + \mathrm{KL}(q_{\boldsymbol{\phi}}(z)||p(z|\hat{\boldsymbol{Y}})). \tag{5.8}$$

It is found that the above variational objective shares the similarity with conditional VAE (CVAE) [150] which is able to produce diverse outputs for a single input. For example, Walker *et al.* [169] use a fully convolutional CVAE for diverse motion prediction from a static image. Zhou and Berg [204] generate diverse time-lapse videos by incorporating conditional,

twostack and recurrent architecture modifications to standard generative models. Therefore, our weakly-supervised VESD naturally embeds the diversity in video summary generation.

5.4.1 Learnable Prior and Posterior

In contrast to the standard VAE prior that assumes the latent variable z to be drawn from latent Gaussian (*e.g.*, $p(z) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$), we impose the prior distribution learned from web videos which infers the topic-specific semantics more accurately. Thus we impose z to be drawn from the Gaussian with $p(z|\hat{\boldsymbol{Y}}) = \mathcal{N}(z|\boldsymbol{\mu}(\hat{\boldsymbol{Y}}), \boldsymbol{\sigma}^2(\hat{\boldsymbol{Y}})\boldsymbol{I})$ whose mean and variance are defined as:

$$\boldsymbol{\mu}(\hat{\boldsymbol{Y}}) = f_{\mu}(\hat{\boldsymbol{Y}}), \log\boldsymbol{\sigma}^2(\hat{\boldsymbol{Y}}) = f_{\sigma}(\hat{\boldsymbol{Y}}), \tag{5.9}$$

where $f_{\mu}(\cdot)$ and $f_{\sigma}(\cdot)$ denote any type of neural networks that are suitable for the observed data. We adopt two-layer MLPs with ReLU activation in our implementation.

Likewise, we model the posterior of $q_{\boldsymbol{\phi}}(z|\cdot) := q_{\boldsymbol{\phi}}(z|\boldsymbol{X}, \hat{\boldsymbol{Y}}, \boldsymbol{c})$ with the Gaussian distribution $\mathcal{N}(z|\boldsymbol{\mu}(\boldsymbol{X}, \hat{\boldsymbol{Y}}, \boldsymbol{c}), \boldsymbol{\sigma}^2(\boldsymbol{X}, \hat{\boldsymbol{Y}}, \boldsymbol{c})$ whose mean and variance are also characterized by two-layer MLPs with ReLU activation:

$$\boldsymbol{\mu} = f_{\mu}([\boldsymbol{a}; \boldsymbol{h}^d(\hat{\boldsymbol{Y}}); \boldsymbol{c}]), \log\boldsymbol{\sigma}^2 = f_{\sigma}([\boldsymbol{a}; \boldsymbol{h}^d(\hat{\boldsymbol{Y}}); \boldsymbol{c}]). \tag{5.10}$$

5.4.2 Mixed Training Objective Function

One potential issue of purely weakly-supervised VESD training objective (5.8) is that the semantic matching loss usually results in summaries focusing on very few shots in raw video. To ensure the diversity and fidelity of the generated summaries, we can also make use of the importance scores on partially finely-annotated benchmark datasets to consistently improves performance. For those detailed annotations in benchmark datasets, we adopt the same keyframe regularizer in [100] to measure the cross-entropy loss between the normalized ground-truth importance scores $\boldsymbol{\alpha}_{\boldsymbol{X}}^{gt}$ and the output attention scores $\boldsymbol{\alpha}_{\boldsymbol{X}}$ as below:

$$\mathcal{L}_{\text{score}} = \text{cross-entropy}(\boldsymbol{\alpha}_{\boldsymbol{X}}^{gt}, \boldsymbol{\alpha}_{\boldsymbol{X}}). \tag{5.11}$$

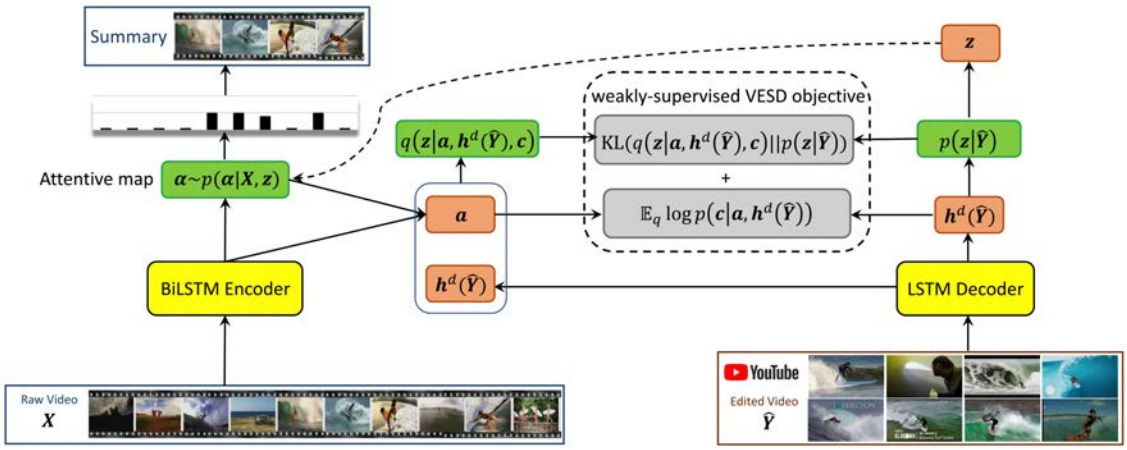


Figure 5.2 The variational formulation of our weakly-supervised VESD framework.

Accordingly, we train the regularized VESD using the following objective function to utilize different levels of annotations:

$$\mathcal{L}_{\text{mixed}} = \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\omega}) + \lambda \mathcal{L}_{\text{score}}. \tag{5.12}$$

The overall objective can be trained using back-propagation efficiently and is illustrated in Fig. 5.2. After training, we calculate the salience score $\boldsymbol{\alpha}$ for each new video by forward passing the summarization model in VESD.

5.5 Experimental Results

Datasets and Evaluation. We test our VESD framework on two publicly available video summarization benchmark datasets CoSum [23] and TVSum [151]. The CoSum [23] dataset consists of 51 videos covering 10 topics including Base Jumping (BJ), Bike Polo (BP), Eiffel Tower (ET), Excavators River Cross (ERC), Kids Playing in leaves (KP), MLB, NFL, Notre Dame Cathedral (NDC), Statue of Liberty (SL) and SurFing (SF). The TVSum [151] dataset contains 50 videos organized into 10 topics from the TRECVid Multimedia Event Detection task [148], including changing Vehicle Tire (VT), getting Vehicle Unstuck (VU), Grooming

an Animal (GA), Making Sandwich (MS), ParKour (PK), PaRade (PR), Flash Mob gathering (FM), BeeKeeping (BK), attempting Bike Tricks (BT), and Dog Show (DS). Following the literature [56, 197], we randomly choose 80% of the videos for training and use the remaining 20% for testing on both datasets. As recommended by [23, 119, 120], we evaluate the quality of a generated summary by comparing it to multiple user-annotated summaries provided in benchmarks. Specifically, we compute the pairwise average precision (AP) for a proposed summary and all its corresponding human-annotated summaries, and then report the mean value. Furthermore, we average over the number of videos to achieve the overall performance on a dataset. For the CoSum dataset, we follow [119, 120] and compare each generated summary with three human-created summaries. For the TVSum dataset, we first average the frame-level importance scores to compute the shot-level scores, and then select the top 50% shots for each video as the human-created summary. Finally, each generated summary is compared with twenty human-created summaries. The top-5 and top-15 mAP performances on both datasets are presented in evaluation.

Web Video Collection. This section describes the details of web video collection for our approach. We treat the topic labels in both datasets as the query keywords and retrieve videos from YouTube for all the twenty topic categories. We limit the videos by time duration (less than 4 minutes) and rank by relevance to constructing a set of weakly-annotated videos. However, these downloaded videos are still very lengthy and noisy in general since they contain a proportion of frames that are irrelevant to search keywords. Therefore, we introduce a simple but efficient strategy to filter out the noisy parts of these web videos: (1) we first adopt the existing temporal segmentation technique KTS [127] to segment both the benchmark videos and web videos into non-overlapping shots, and utilize CNNs to extract feature within each shot; (2) the corresponding features in benchmark videos are then used to train a MLP with their topic labels (the shots do not belong to any topic label are set with background label) and perform prediction for the shots in web videos; (3) we further truncate web videos based on

the relevant shots whose topic-related probability is larger than a threshold. In this way, we observe that the trimmed videos are sufficiently clean and informative for learning the latent semantics in our VAE module.

Architecture and Implementation Details. For the fair comparison with state-of-the-art methods [100, 197], we choose to use the output of pool5 layer of the GoogLeNet [158] for the frame-level feature. The shot-level feature is then obtained by averaging all the frame features within a shot. We first use the features of segmented shots on web videos to pre-train a VAE module whose dimension of the latent variable is set to 256. To build encoder-summarizer-decoder, we use a two-layer bidirectional LSTM with 1024 hidden units, a two-layer MLP with [256, 256] hidden units and a two-layer LSTM with 1024 hidden units for the encoder RNN, attention MLP and decoder RNNs, respectively. For the parameter initialization, we train our framework from scratch using stochastic gradient descent with a minibatch size of 20, a momentum of 0.9, and a weight decay of 0.005. The learning rate is initialized to 0.01 and is reduced to its 1/10 after every 20 epochs (100 epochs in total). The trade-off parameter λ is set to 0.2 in the mixed training objective.

5.5.1 Quantitative Results

Exploration Study. To better understand the impact of using web videos and different types of annotations in our method, we analyzed the performances under the following six training settings: (1) benchmark datasets with weak supervision (topic labels); (2) benchmark datasets with weak supervision and extra 30 downloaded videos per topic; (3) benchmark datasets with weak supervision and extra 60 downloaded videos per topic; (4) benchmark datasets with strong supervision (topic labels and importance scores); (5) benchmark datasets with strong supervision and extra 30 downloaded videos per topic; and (6) benchmark datasets with strong supervision and extra 60 downloaded videos per topic. We have the following key observations from Table 5.1: (1) Training on the benchmark data with only weak topic labels in our VESD

Table 5.1 Exploration study on training settings. Numbers show top-5 mAP scores.

Training Settings	CoSum	TVSum
benchmark with weak supervision	0.616	0.352
benchmark with weak supervision + 30 web videos/topic	0.684	0.407
benchmark with weak supervision + 60 web videos/topic	0.701	0.423
benchmark with strong supervision	0.712	0.437
benchmark with strong supervision + 30 web videos/topic	0.755	0.481
benchmark with strong supervision + 60 web videos/topic	0.764	0.498

Table 5.2 Performance comparison using different types of features on CoSum dataset. Numbers show top-5 mAP scores averaged over all the videos of the same topic.

Features	BJ	BK	ET	ERC	KP	MLB	NFL	NDC	SL	SF	Top-5
GoogLeNet	0.715	0.746	0.813	0.756	0.772	0.727	0.737	0.782	0.794	0.709	0.755
ResNet101	0.727	0.755	0.827	0.766	0.783	0.741	0.752	0.790	0.807	0.722	0.767
C3D	0.729	0.754	0.831	0.761	0.779	0.740	0.747	0.785	0.805	0.718	0.765

framework performs much worse than either that of training using extra web videos or that of training using detailed importance scores, which demonstrates our generative summarization model demands a larger amount of annotated data to perform well. (2) We notice that the more web videos give better results, which clearly demonstrates the benefits of using web videos and proves the scalability of our generative framework. (3) This big improvements with strong supervision illustrate the positive impact of incorporating available importance scores for mixed training of our VESD. That is not surprising since the attention scores should be imposed to focus on different fragments of raw videos in order to be consistent with ground-truths, resulting in the summarizer with the diverse property which is an important metric in generating good summaries. We use the training setting (5) in the following experimental comparisons.

Effect of Deep Features. We also investigate the effect of using different types of deep fea-

Table 5.3 Experimental results on CoSum dataset. Numbers show top-5/15 mAP scores averaged over all the videos of the same topic.

Topic	Unsupervised Methods					Supervised Methods					VESD
	SMRS	Quasi	MBF	CVS	SG	KVS	DPP	sLstm	SM	DSN	
BJ	0.504	0.561	0.631	0.658	0.698	0.662	0.672	0.683	0.692	0.685	**0.715**
BP	0.492	0.625	0.592	0.675	0.713	0.674	0.682	0.701	0.722	0.714	**0.746**
ET	0.556	0.575	0.618	0.722	0.759	0.731	0.744	0.749	0.789	0.783	**0.813**
ERC	0.525	0.563	0.575	0.693	0.729	0.685	0.694	0.717	0.728	0.721	**0.756**
KP	0.521	0.557	0.594	0.707	0.729	0.701	0.705	0.714	0.745	0.742	**0.772**
MLB	0.543	0.563	0.624	0.679	0.721	0.668	0.677	0.714	0.693	0.687	**0.727**
NFL	0.558	0.587	0.603	0.674	0.693	0.671	0.681	0.681	0.727	0.724	**0.737**
NDC	0.496	0.617	0.595	0.702	0.738	0.698	0.704	0.722	0.759	0.751	**0.782**
SL	0.525	0.551	0.602	0.715	0.743	0.713	0.722	0.721	0.766	0.763	**0.794**
SF	0.533	0.562	0.594	0.647	0.681	0.642	0.648	0.653	0.683	0.674	**0.709**
Top-5	**0.525**	**0.576**	**0.602**	**0.687**	**0.720**	**0.684**	**0.692**	**0.705**	**0.735**	**0.721**	**0.755**
Top-15	**0.547**	**0.591**	**0.617**	**0.699**	**0.731**	**0.702**	**0.711**	**0.717**	**0.746**	**0.736**	**0.764**

tures as shot representation in VESD framework, including 2D deep features extracted from GoogLeNet [158] and ResNet101 [61], and 3D deep features extracted from C3D [163]. In Table 5.2, we have following observations: (1) ResNet produces better results than GoogLeNet, with a top-5 mAP score improvement of 0.012 on the CoSum dataset, which indicates more powerful visual features still lead improvement for our method. We also compare 2D GoogLeNet features with C3D features. Results show that the C3D features achieve better performance over GoogLeNet features (0.765 vs 0.755) and comparable performance with ResNet101 features. We believe this is because C3D features exploit the temporal information of videos thus are also suitable for summarization.

Comparison with Unsupervised Methods. We first compare VESD with several unsupervised methods including SMRS [33], Quasi [79], MBF [23], CVS [120] and SG [100]. Table. 5.3 shows the mean AP on both top 5 and 15 shots included in the summaries for the Co-

Table 5.4 Experimental results on TVSum dataset. Numbers show top-5/15 mAP scores averaged over all the videos of the same topic.

Topic	Unsupervised Methods					Supervised Methods					VESD
	SMRS	Quasi	MBF	CVS	SG	KVS	DPP	sLstm	SM	DSN	
VT	0.272	0.336	0.295	0.328	0.423	0.353	0.399	0.411	0.415	0.373	**0.447**
VU	0.324	0.369	0.357	0.413	0.472	0.441	0.453	0.462	0.467	0.441	**0.493**
GA	0.331	0.342	0.325	0.379	0.475	0.402	0.457	0.463	0.469	0.428	**0.496**
MS	0.362	0.375	0.412	0.398	0.489	0.417	0.462	0.477	0.478	0.436	**0.503**
PK	0.289	0.324	0.318	0.354	0.456	0.382	0.437	0.448	0.445	0.411	**0.478**
PR	0.276	0.301	0.334	0.381	0.473	0.403	0.446	0.461	0.458	0.417	**0.485**
FM	0.302	0.318	0.365	0.365	0.464	0.397	0.442	0.452	0.451	0.412	**0.487**
BK	0.297	0.295	0.313	0.326	0.417	0.342	0.395	0.406	0.407	0.368	**0.441**
BT	0.314	0.327	0.365	0.402	0.483	0.419	0.464	0.471	0.473	0.435	**0.492**
DS	0.295	0.309	0.357	0.378	0.466	0.394	0.449	0.455	0.453	0.416	**0.488**
Top-5	**0.306**	**0.329**	**0.345**	**0.372**	**0.462**	**0.398**	**0.447**	**0.451**	**0.461**	**0.424**	**0.481**
Top-15	**0.328**	**0.347**	**0.361**	**0.385**	**0.475**	**0.412**	**0.462**	**0.464**	**0.483**	**0.438**	**0.503**

Sum dataset, whereas Table 5.4 shows the results on TVSum dataset. We can observe that: (1) Our weakly supervised approach obtains the highest overall mAP and outperforms traditional non-DNN based methods SMRS, Quasi, MBF and CVS by large margins. (2) The most competing DNN based method, SG [100] gives top-5 mAP that is 3.5% and 1.9% less than ours on the CoSum and TVSum dataset, respectively. Note that with web videos only is better than training with multiple handcrafted regularizations proposed in SG. This confirms the effectiveness of incorporating a large number of web videos in our framework and learning the topic-specific semantics using a weakly-supervised matching loss function. (3) Since the CoSum dataset contains videos that have visual concepts shared with other videos from different topics, our approach using generative modelling naturally yields better results than that on the TVSum dataset. (4) It's worth noticing that TVSum is a quite challenging summarization dataset because topics on this dataset are very ambiguous and difficult to understand well with

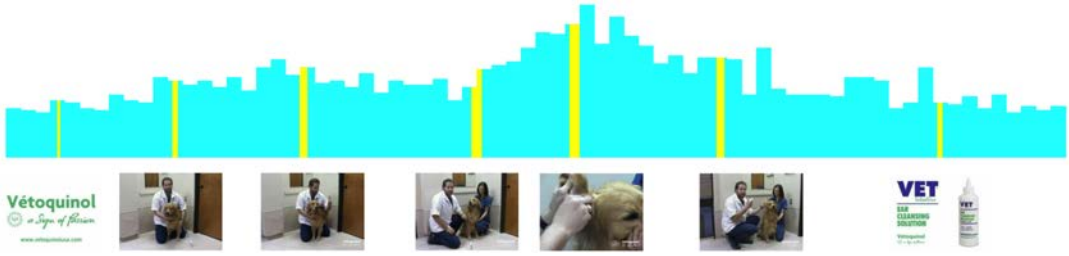
very few videos. By accessing the similar web videos to eliminate ambiguity for a specific topic, our approach works much better than all the unsupervised methods by achieving a top-5 mAP of 48.1%, showing that the accurate and user-interested video contents can be directly learned from more diverse data rather than complex summarization criteria.

Comparison with Supervised Methods. We then conduct comparison with some supervised alternatives including KVS [127], DPP [46], sLstm [197], SM [56] and DSN [119] (weakly-supervised), we have the following key observations from Table. 5.3 and Table. 5.4: (1) VESD outperforms KVS on both datasets by a big margin (maximum improvement of 7.1% in top-5 mAP on CoSum), showing the advantage of our generative modelling and more powerful representation learning with web videos. (2) On the Cosum dataset, VESD outperforms SM [56] and DSN [119] by a margin of 2.0% and 3.4% in top-5 mAP, respectively. The results suggest that our method is still better than the fully-supervised methods and the weakly-supervised method. (3) On the TVSum dataset, a similar performance gain of 2.0% can be achieved compared with all other supervised methods.

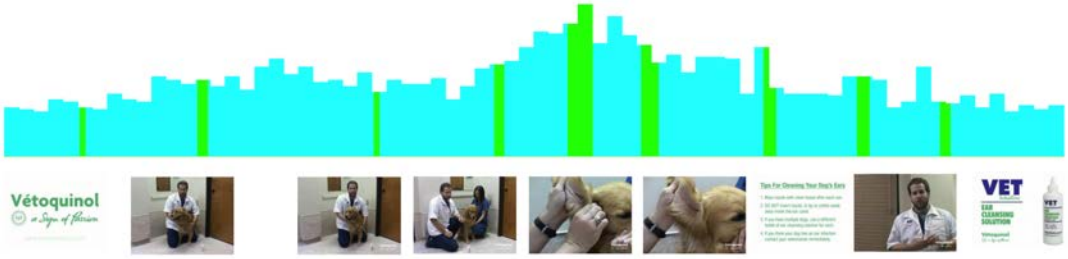
5.5.2 Qualitative results

To get some intuition about the different training settings for VESD and their effects on the temporal selection pattern, we visualize some selected frames on an example video in Fig. 5.3. The cyan background shows the frame-level importance scores. The coloured regions are the selected subset of frames using the specific training setting. The visualized keyframes for different setting supports the results presented in Table 5.1. We notice that all four settings cover the temporal regions with the high frame-level score. In the last subfigure, we can easily see that weakly-supervised VESD with web videos and available importance scores produces more reliable summaries than training on benchmark videos with only weak labels. That is, by leveraging both the web videos and importance scores in datasets, VESD framework will shift towards the highly topic-specific temporal regions.

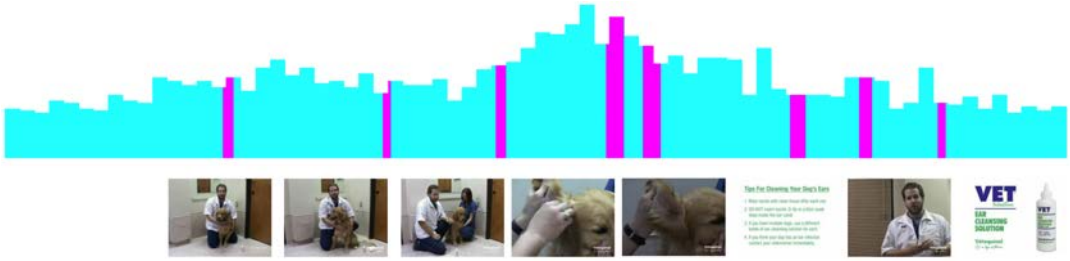
(a) Sample frames from video 15 [151]



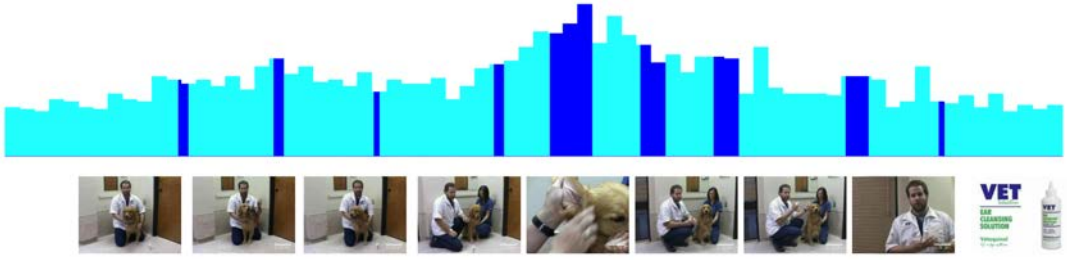
(b) Training on benchmark with weak supervision



(c) Training on benchmark with weak supervision and extra web videos



(d) Training on benchmark with strong supervision



(e) Training on benchmark with strong supervision and extra web videos

Figure 5.3 Qualitative comparison of video summaries using different training settings, along with the ground-truth importance scores (cyan background). (Best viewed in colors)

5.6 Conclusion

One key problem in video summarization is how to model the latent semantic representation, which has not been adequately resolved under the ”single video understanding” framework in prior works. To address this issue, we introduced a generative summarization framework called VESD to leverage the web videos for better latent semantic modelling and to reduce the ambiguity of video summarization in a principled way. We incorporated flexible web prior distribution into a variational framework and presented a simple encoder-decoder with attention for summarization. The potentials of our VESD framework for large-scale video summarization were validated, and extensive experiments on benchmarks showed that VESD outperforms state-of-the-art video summarization methods significantly.

Chapter 6

Learning A Structured Network for Discriminative Centralized Sparse Representations

6.1 Introduction

Recent advances on training deep architectures such as CNNs have shown leading performance to learn discriminative feature representations in a variety of computer vision and machine learning problems. Though CNNs are very powerful to learn from large-scale datasets [61, 135, 146], they have two limitations in visual recognition tasks with limited training data such as FGVC and texture classification. First, the widely-used fine-tuned networks [146] for these visual recognition tasks are usually unstructured and over-parameterized [25, 94], raising concerns on model overfitting and effectiveness. Second, the availability of training data in existing benchmarks of those applications is often very scarce to reliably represent the data distribution of each category in the feature space; therefore, the learning of distinct patterns and group structures becomes very difficult.

To overcome those difficulties, many works have been recently done to learn discriminative representations with deep hybrid architectures by introducing structured modules on top of CNNs [5, 25, 25, 93, 94, 193, 199]. For example, in FGVC, various part-based approaches [193, 199] have been proposed to capture the subtle local structure and discriminate between neighboring classes. The progress in texture classification has been focusing on the feature encoding techniques [25, 93] to capture local invariance of texture structure. Although these efforts on structured modelling have achieved noticeable performance boost, they either demand finer-level but costly annotations (*i.e.*, part bounding boxes in fine-grained classification) or depend on redundant encoding networks with a large number of parameters.

Different from previous structured modelling approaches which are mostly customized to specific tasks and involve expensive/complex inference, we propose to embed the classical sparse models [177, 185] into CNNs to build a structured network. Our idea is motivated by the following facts. First, sparse models are appealing structured techniques because of their ability to learn discriminative subspaces with strong interpretability. Second, sparse models are preferable in medium-sized recognition regime as they need much less training data than modern CNNs with significantly fewer model parameters. Third, sparse models usually rely on iterative approximation algorithms, whose inherently sequential structure and complexity constitute a major bottleneck in the computational efficiency. Fortunately, from the viewpoint of recurrent neural networks, sparse models can be approximated by unfolding and truncating the iterative optimization algorithms so that end-to-end training with efficiently computational blocks can be enabled in a weight-sharing manner [52, 153].

By exploring the above merits of sparse models, we propose a structured network that exhibits desired properties of class-discriminative feature representation, compact network architecture and learning efficiency. Specifically, we enforce the sparse model to have small intra-class variation so that discriminative centralized sparse representations (DCSR) will be generated, and further reformulate the sparse model as a feed-forward network, namely DCSR-Net,

which involves only a small number of parameters. The DCSR-Net can be easily embedded on top of many existing CNN architectures, acting as a structured regularization network to improve the generalization performance of CNNs, particularly for those recognition tasks with limited training data. Our technical merits can be summarized as follows: First, we propose a structured network that can be plugged into CNNs as an effective structure-aware regularization. It allows producing reliable discriminative deep representations for limited-sized visual tasks. Second, the proposed DCSR-Net is derived from the learning framework of sparse models. It is able to provide an efficient inference process with negligible parameter complexity and thus can be regarded as costless structured modelling. Third, extensive experiments on FGVC and texture classification benchmarks demonstrate that, coupling DCSR-Net with CNNs achieves the significant performance boost.

6.2 Related Work

6.2.1 Deep Structured Unrolling Models

Deep structured models aim to model structured patterns within off-the-shelf building blocks in DNNs. Among the many attempts, a noticeable portion of efforts has been devoted to unrolling the traditional optimization and inference algorithms into a deep feed-forward structure, which enables end-to-end training. In the pioneer work [52], a learning framework of the iterative shrinkage-thresholding algorithm (LISTA) is proposed to efficiently approximate the desired sparse codes by incorporating the problem/data structure into the design of deep architectures, demonstrating benefits in both performance and interpretability. It is also demonstrated in [180] that a DNN can recover ℓ_0-norm sparse representations [174] under mild theoretical conditions. In [172], the proximal methods are introduced to deep models with continuous output variables. More examples include shrinkage fields [137], CRF-RNN [202], and ADMM-net [155]. By turning optimization algorithms into DNNs, one may expect faster

inference, better scalability and most importantly, more elaborate structures. Different from the aforementioned works in the scenario of unsupervised learning, in this work we attempt to learn discriminative structures from data via supervised sparse models.

6.2.2 Structured Representation Learning with Loss Functions

There have been blooming interests in training DNNs jointly with new loss functions to improve the model generalization performance. The idea is to regard loss functions as structured regularizations and learn feature representations to meet the pre-defined structural priors. In [31], the linear discriminant analysis is directly translated as a training criterion for DNNs. In [138], the triplet loss is employed to minimize distances of sample features from the same class while maximizing distances of samples of different classes. Other loss functions include center loss [176] that embeds the class-oriented clustering structure, and angular softmax loss [96] that exploits the large-margin separate structure.

Our approach differs from the aforementioned methods in that we employ a structured network instead of loss functions to encode latent features and train the network in a supervised end-to-end fashion. Our approach allows a much broader choice for structured modelling in network architecture and further improves the performance.

6.3 DCSR Model

6.3.1 From Loss Function to Structured Module

To improve the model generalization performance for recognition tasks with limited training data, many previous works equipped with deep architectures employ a joint objective function consisting of both the softmax classifier and some structured loss function. Denoting by $h_{\boldsymbol{\Theta}}(\boldsymbol{x})$ the deep feature of input sample $\boldsymbol{x}$ via feature extractor $h_{\boldsymbol{\Theta}}(\cdot)$ (*e.g.*, CNNs) parameterized by

$\boldsymbol{\Theta}$. The general objective function to be minimized in network training can be written as:

$$\mathcal{L}_f = \underbrace{\frac{1}{|\mathcal{X}|}\sum_{(\boldsymbol{x},y)} \ell_{so}(y, h_{\boldsymbol{\Theta}}(\boldsymbol{x}); \boldsymbol{\Psi})}_{\text{softmax loss } \mathcal{L}_{so}} + \underbrace{\frac{\lambda}{|\mathcal{X}|}\sum_{(\boldsymbol{x},y)} \ell_{st}(y, h_{\boldsymbol{\Theta}}(\boldsymbol{x}); \boldsymbol{\Phi})}_{\text{structured loss } \mathcal{L}_{st}}, \tag{6.1}$$

where $(\boldsymbol{x}, y) \in (\mathcal{X}, \mathcal{Y})$ refers to the training sample and its label, $\boldsymbol{\Theta}$, $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ denote the parameter sets of feature extractor $h(\cdot)$ (*e.g.*, CNNs), softmax regressor $\ell_{so}(\cdot)$ and structured term $\ell_{st}(\cdot)$, respectively, while parameter λ is to balance softmax loss $\mathcal{L}_{so}$ and structured loss $\mathcal{L}_{st}$. Some recent structured losses have been introduced to reduce the intra-class variations. In [149], prototypical networks are developed to take a class prototype as the mean of its support set in the embedding space in order to address the overfitting issue in few-shot learning. In [176], the center loss is proposed to decrease the distances between samples and their class centers to make the learned face representations more discriminative.

Different from previous approaches which usually minimize a structured loss as given in Eqn. (6.1), we propose to reformulate the structured loss as a feed-forward structured module built upon the hidden variables in high-level layer of DNNs. Instead of joint training with softmax loss and some structured loss, we construct a hierarchical architecture by passing $h_{\boldsymbol{\Theta}}(\cdot)$ through a learnable structured module, denoted by $g^{st}_{\boldsymbol{\Phi}}(\cdot)$ (parameterized by $\boldsymbol{\Phi}$), ahead of the softmax classifier. Mathematically, our proposed network architecture can be described as:

$$\mathcal{L}_f = \frac{1}{|\mathcal{X}|}\sum_{(\boldsymbol{x},y)} \ell_{so}(y, g^{st}_{\boldsymbol{\Phi}}(h_{\boldsymbol{\Theta}}(\boldsymbol{x})); \boldsymbol{\Psi}). \tag{6.2}$$

Compared with the conventional structured loss based network training in Eqn. (6.1), the learning in our proposed architecture is mainly accomplished via the structured module $g^{st}_{\boldsymbol{\Phi}}(\cdot)$, with which the network could adaptively exploit the potential inter/intra-class data structure, and consequently make the learned representations discriminative. Denote by $\boldsymbol{\Gamma} = \{\boldsymbol{\Theta}, \boldsymbol{\Psi}, \boldsymbol{\Phi}\}$ the set of parameters in Eqn. (6.2). Note that the learning of our module requires the computation of gradient $\nabla_{\boldsymbol{\Gamma}} \ell_{so}(\cdot)$, which in turn relies on the gradient of $\boldsymbol{z}_{\boldsymbol{\Phi}} := g^{st}_{\boldsymbol{\Phi}}(\cdot)$ with respect

to $\boldsymbol{\Phi}$. Therefore, one key issue is how to construct a well-defined $g^{st}_{\boldsymbol{\Phi}}(\cdot)$ that enables elaborate structure, fast inference and end-to-end learning. Considering that sparse coding or sparse representation is an effective and flexible technique for structured modeling, in the following sections we present a discriminative sparse coding model as the structured module $g^{st}_{\boldsymbol{\Phi}}(\cdot)$ to encourage more discriminative and clustered representations, and then present how to embed it in DNNs.

6.3.2 Discriminative Sparse Model as Structured Module

As discussed above, our hierarchical architecture aims to implement a discriminative sparse model as the structured module $g^{st}_{\boldsymbol{\Phi}}(\cdot)$ to more discriminatively encode the high-level features for classification. The parameter set $\boldsymbol{\Phi}$ of the structured module can be understood as the dictionary in sparse models. However, typical sparse models are generative architectures that generate the sparse representations from the perspective of reconstruction, while they have limited capability in grouping label-consistent samples to share similar representations. To endow the sparse model class-aware structures in supervised learning tasks, we propose a discriminative centralized sparse representation (DCSR) model. Refer to Eqn. (6.2), we denote by $\boldsymbol{h} := h_{\boldsymbol{\Theta}}(\boldsymbol{x})$ the high level features output by the previous layers of a network. The DCSR model is formulated as follows:

$$\boldsymbol{z}_{\boldsymbol{\Phi}} = \arg\min_{\boldsymbol{z},\boldsymbol{\Phi}} \|\boldsymbol{h} - \boldsymbol{D}\boldsymbol{z}\|_2^2 + \delta\|\boldsymbol{z} - \boldsymbol{m}_y\|_2^2 + \tau\|\boldsymbol{z}\|_1, \tag{6.3}$$

where $\boldsymbol{D}$ denotes the dictionary with $\|\boldsymbol{D}\|_2 = 1$ by default; $\boldsymbol{m}_y$ is the mean vector of all sparse codes $\boldsymbol{z}$ for class k if $y = k$; $\boldsymbol{\Phi} := \{\boldsymbol{D}, \boldsymbol{M}\}$, where $\boldsymbol{M} = \{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K\}$, is the set of parameters for the structured module; δ and τ are trade-off weights.

Based on the above DCSR model, the feature representations $\boldsymbol{z}_{\boldsymbol{\Phi}}$ are optimized to push examples from the same class closer to its class center. Notice that when $\delta, \tau > 0$, Eqn. (6.3) is strictly convex given $\boldsymbol{\Phi}$, and consequently, $\boldsymbol{z}_{\boldsymbol{\Phi}}$ defines an unambiguous deterministic map from the data space to the space of the class centralized sparse representations. More specifically,

the optimal representation can be derived as the fixed point of the following equation:

$$\boldsymbol{z} = s_{\frac{\tau}{\beta}}(\boldsymbol{z} - \frac{1}{\beta}[2\boldsymbol{D}^T(\boldsymbol{D}\boldsymbol{z} - \boldsymbol{h}) + 2\delta(\boldsymbol{z} - \boldsymbol{m}_y)]), \tag{6.4}$$

where $s_\theta = \text{sign}(z)(|z| - \theta)_+$ denotes the element-wise soft thresholding and β is a constant that defines an upper bound on the largest eigenvalue of $\boldsymbol{D}^T\boldsymbol{D}$. One can also compute explicitly the gradient with respect to $\boldsymbol{\Phi} := \{\boldsymbol{D}, \boldsymbol{M}\}$, and $\boldsymbol{z}_{\boldsymbol{\Phi}}$ provides a desired form of structured module $g^{st}_{\boldsymbol{\Phi}}(\cdot)$.

Though sparse modelling provides a sophisticated and analytical way to build structured data models, the exact gradient computation in Eqn. (6.3) is usually complex and has relatively high computational complexity and latency. The iterative optimization scheme of Eqn. (6.4) greatly depends on the given problem and usually provides worst-case (data-independent) convergence rate to explore the intrinsic property, *e.g.*, low-dimensional manifold, of data. Such a discrepancy hinders the computational efficiency improvement in deep architectures. In the following, therefore, we present a fast trainable framework that implements the DCSR model very effectively and efficiently.

6.4 DCSR Driven Network

6.4.1 Trainable DCSR Model

From the perspective of iterative optimization, Eqn. (6.3) is merely a proxy to obtain a non-linear mapping between the feature $\boldsymbol{h}$ and the discriminative centralized representation $\boldsymbol{z}_{\boldsymbol{\Phi}}$. The mapping (6.4) can be expressed by unrolling a sufficient number T of iterations into a feed-forward network comprising T (identical) layers. However, in practice the complexity budget may require T to be a small fixed number, leading to an unsatisfactory representation $\boldsymbol{z}_{\boldsymbol{\Phi}}$. We define the following parameters $\boldsymbol{H}, \boldsymbol{W}, \{\boldsymbol{c}_y\}, \boldsymbol{\theta}$ for Eqn. (6.3):

$$\boldsymbol{H} = \frac{\beta - 2\delta}{\beta}\boldsymbol{I} - \frac{2}{\beta}\boldsymbol{D}^T\boldsymbol{D}, \boldsymbol{W} = \frac{2}{\beta}\boldsymbol{D}^T, \boldsymbol{c}_y = \frac{2\delta}{\beta}\boldsymbol{m}_y, \boldsymbol{\theta} = \frac{\tau}{\beta}. \tag{6.5}$$

Algorithm 6.4.1: Backpropagation process for the computation of the sub-gradients of $\ell(z)$. $\delta*$ denotes the gradient of ℓ with respect to $*$ as customary in neural network literature. $\odot$ denotes element-wise product. s'_{θ} denotes the jacobian of s with respect to its input.

input: Sub-gradient δz of ℓ with respect to network output; intermediate layer outputs

output: Sub-gradients of ℓ with respect to the parameters, $\delta \boldsymbol{H}, \delta \boldsymbol{W}, \{\delta \boldsymbol{c}_k\}, \delta \boldsymbol{\theta}$.

Initialize $\delta \boldsymbol{H} = 0; \delta \boldsymbol{W} = 0; \delta \boldsymbol{\theta} = 0, \delta z_T = 0$

for $t = T$ down to 1 **do**

$\delta \boldsymbol{a}_t = s'_{\boldsymbol{\theta}}(\boldsymbol{a}_t) \odot \delta \boldsymbol{z}_t$

$\delta \boldsymbol{\theta} = \delta \boldsymbol{\theta} - \text{sign}(\boldsymbol{a}_t) \odot \delta \boldsymbol{a}_t$

$\delta \boldsymbol{W} = \delta \boldsymbol{W} + \delta \boldsymbol{a}_t \boldsymbol{z}_{t-1}^T$

$\delta \boldsymbol{H} = \delta \boldsymbol{H} + \delta \boldsymbol{a}_t \boldsymbol{z}_{t-1}^T$

$\delta \boldsymbol{c}_k = \frac{\sum_{(\boldsymbol{x},y)} \mathbf{1}\{y=k\} \odot (\boldsymbol{c}_k - \boldsymbol{h})}{1 + \sum_{(\boldsymbol{x},y)} \mathbf{1}\{y=k\}}$

$\delta \boldsymbol{z}_{t-1} = \boldsymbol{H}^T \delta \boldsymbol{a}_t$

end

The above newly defined parameters can be collectively denoted as another form of $\boldsymbol{\Phi}$ which has larger model capacity and easier training in terms of neural network.

Within the family of identical inference layers producing representation $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$, which refers to the step-T truncating of $\boldsymbol{z}_{\boldsymbol{\Phi}}$, there might exist better parameters with which $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$ performs better on the given data. Such parameters can be obtained via learning from data. In this manner, we can obtain a complexity fixed DCSR model $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$ as the structured module $g^{st}_{\boldsymbol{\Phi}}(\cdot)$. Similar idea was first advocated in [52], where Gregor and LeCun unrolled the standard iterative shrinkage-thresholding algorithm (ISTA) into a fixed-depth DNNs and learned a new set of parameters to approximate the optimal sparse codes with small computational cost. This approach was later extended to more elaborated structured sparse and low-rank models [153].

Then the inference process in $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$ is reformulated as the following iterative rule:

$$\boldsymbol{z}_{t+1} = s_{\boldsymbol{\theta}}(\boldsymbol{H}\boldsymbol{z}_t + \boldsymbol{W}\boldsymbol{h} + \boldsymbol{c}_{y_x}). \tag{6.6}$$

The forward propagation of $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$ is straightforward and it is depicted as the block diagram in the upper rectangle box in Fig. 6.1. The learning of parameters $\boldsymbol{\Phi}$ requires computing the sub-gradients $d\ell(\boldsymbol{z})/d\boldsymbol{\Phi}$, which is accomplished by the back-propagation procedure by applying the chain rule. Back-propagation starts with differentiating $\ell(\boldsymbol{z})$ with respect to the output of the last network layer, and propagating the sub-gradients down to the input layer. The complete back-propagation procedure is summarized in Algorithm 6.4.1.

6.4.2 Siamese Architecture of DCSR Network

Note that the inference process of the trainable DCSR model in Eqn. (6.6) requires input $(\boldsymbol{x}, y)$ in the training stage. However, the label y is unavailable in the testing phase, making it not directly applicable to prediction. This problem can be easily solved by learning a siamese architecture simultaneously for the trainable DCSR model, resulting in our final DCSR driven network, called DCSR-Net.

Specifically, DCSR-Net jointly learns an unsupervised network branch which approximates the discriminative representations obtained by the trainable DCSR model. The overall architecture of DCSR-Net is illustrated in Fig. 6.1, where we adopt the alternative inference process of $\boldsymbol{z}_{t+1} = ReLU(\boldsymbol{H}\boldsymbol{z}_t + \boldsymbol{W}\boldsymbol{h} + \boldsymbol{c}_{y_x} - \boldsymbol{\theta})$ by imposing the non-negativity on $\hat{\boldsymbol{z}}_{T,\boldsymbol{\Phi}}$. The generated features from the unsupervised branch of DCSR-Net can be used for class label prediction in testing stage. Accordingly, the objective function in 6.2 is transformed into:

$$\mathcal{L}_f = \frac{1}{|\mathcal{X}|}\sum_{(\boldsymbol{x},y)} \ell_{so}(y, g^{st}_{\boldsymbol{\Phi}}(h_{\boldsymbol{\Theta}}(\boldsymbol{x})); \boldsymbol{\Psi}) + \omega\|g^{st}_{\boldsymbol{\Phi}}(h_{\boldsymbol{\Theta}}(\boldsymbol{x})) - u_{\boldsymbol{\Omega}}(h_{\boldsymbol{\Theta}}(\boldsymbol{x}))\|_2^2, \tag{6.7}$$

where we specify $u_{\boldsymbol{\Omega}}$ as the trainable sparse model without the centralized term in Eqn. (6.3), and use the same truncated number of iterations, parameter notation $\boldsymbol{\Omega} := \{\hat{\boldsymbol{H}}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{\theta}}\}$ for structural consistency, where $\hat{\boldsymbol{H}}, \hat{\boldsymbol{W}}$ and $\hat{\boldsymbol{\theta}}$ are defined by setting $\delta = 0$ in Eqn. (6.5) and $\hat{\boldsymbol{z}}_t$ takes

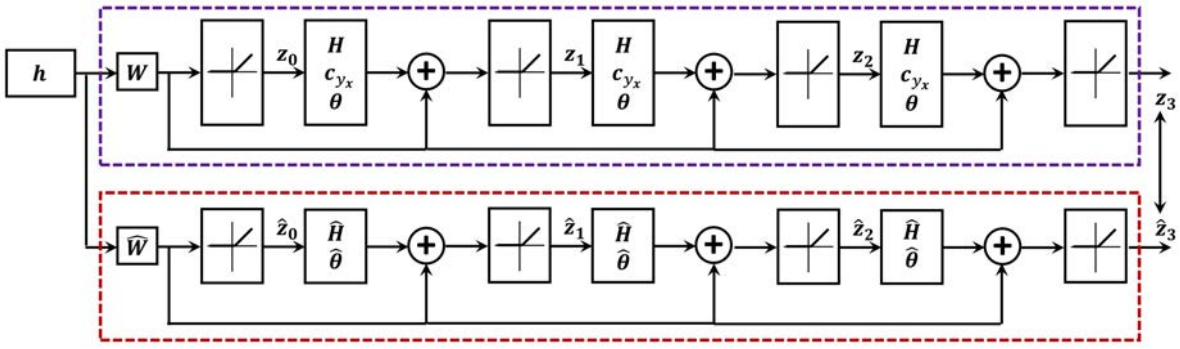


Figure 6.1 DCSR-Net comprises 3 identical layers. The network learns both the DCSR sparse model (purple rectangle box) and standard sparse model (red rectangle box) with a siamese architecture.

the iterative form of Eqn. (6.6) without $\boldsymbol{c}_{y_x}$. The newly-added loss term in Eqn. (6.7) imposes the output of unsupervised sparse model to be identical to that of DCSR model, which can be easily trained and used for classification. The balance weight is set to 0.1 in all our experiments.

In addition, the computation of DCSR-Net is very efficient. It involves only a few matrix multiplications with the inference time complexity of $O(mn + Tm^2)$ (a small $T = 3$ is sufficient in our experiments), where n, m and T is the dimension of $\boldsymbol{h}$, dimension of $\boldsymbol{z}$ and the number of iterations, respectively.

6.5 Experimental Results

In this section, we first perform a comprehensive exploratory study for DCSR-Net, and then evaluate our approach in comparison with state-of-the-art results on the texture classification and FGVC benchmarks.

Network settings. We use the 16-layer VGG network (VGG-16) [146] trained on ImageNet [135] as the base CNN for all our classification experiments. We deploy our DCSR-Net after performing the global average pooling operation on the last convolutional activation layer (*i.e.*,*relu*5_3) of the VGG-16 network. We set $T = 3$ which corresponds a 3-layer DCSR-Net,

and set the dimension of DCSR representation to 4096.

Training implementations. Following [13, 94], we pre-process the input image by cropping the largest image region around its center, resizing it to 448×448, and creating its mirrored version to double the size of the training set. The parameters in DCSR-Net are randomly initialized, and the learning rates of the layers in VGG-16 and DCSR-Net are set as 0.001 and 0.01, respectively. We train all the networks using SGD-Momentum with a batch size of 16 and momentum of 0.9. The training stops at 40 iterations. In the testing phase, we follow the popular CNN-SVM scheme [94].

Datasets and evaluations. For texture classification, we experiment on two benchmarks - the Describable Texture Dataset (DTD) [24] and KTH-TISP2-b (KTH-T2b) [15]. DTD consists of 5,640 real-world texture images labelled with 47 describable texture attributes. KTH-T2b includes 4,752 images of 11 materials captured under controlled scale, pose, and illumination. On DTD, we use the default training/testing splits, and within each split, 2/3 of the images are used for training and 1/3 for testing. On KTH-T2b, each class has four samples. We follow the standard protocol by training on one sample per class and test on the remaining three samples. On both datasets, four splits of training and testing are conducted, and the average accuracy is used as the performance metric.

For FGVC, we also experiment on two popular benchmarks - Caltech-UCSD Bird-200-2011 (CUB) [167] and Aircraft [104]. CUB dataset contains 11,788 bird images of 200 species. We adopt the publicly available split [167], which uses 5,994 images for training and 5,794 for testing. The Aircraft dataset has 100 different aircraft model variants. We adopt the training/testing split protocol provided by [104].

6.5.1 Exploratory Study

Comparison with unstructured networks. To investigate the merit of DCSR as a structured module, we apply DCSR-Net on the MNIST dataset [65] which contains 60,000 training

Table 6.1 MNIST digit classification results of different networks.

Network	Configuration	#Parameters	Error rate (%)
MLP	3-layer(300+100)	266,200	3.03
	3-layer(500+150)	468,500	2.91
	3-layer(500+300)	545,000	1.45
LISTA	1-iter	65,536	1.63
	5-iter	65,536	1.49
	10-iter	65,536	1.36
DCSR-Net	T=2	132,572	1.94
	T=3	132,572	1.23
	T=4	132,572	1.19

and 10,000 test images. Please note that our goal here is not to obtain state-of-the-art results on MNIST digit classification, but rather to provide a fair analysis of the effectiveness of learning class-aware structures in the context of neural networks, compared with the unstructured networks such as multi-layer perceptron (MLP) with ReLU nonlinearity and LISTA [52]. We adopt DCSR-Net as a feature extractor and then use logistic regression to classify the features. The dimension of the original images is reduced to 128-dim by PCA, and the feature for classification is set to 150-dim vector. We use SGD with weight decay of 0.0002, a momentum of 0.9 and a mini-batch size of 100. The initial learning rate of DCSR-Net is 0.1, decayed to 0.01 at 100 epochs. We augment DCSR-Net with a softmax layer that is randomly initialized. The results are shown in Table 6.1. We observe that DCSR-Net achieves lower error rate than the others. Especially, it takes 10 iterations for LISTA to achieve an error rate of 1.36, but only 3 equivalent layers for DCSR-Net to achieve an error rate of 1.23. Fig. 6.2 presents visualization of the learned features during the interations for DCSR-Net. The images in the first and second row indicate the feature learning process on the training and testing set, respectively. We observe that DCSR-Net results in class-aware feature space and the structured

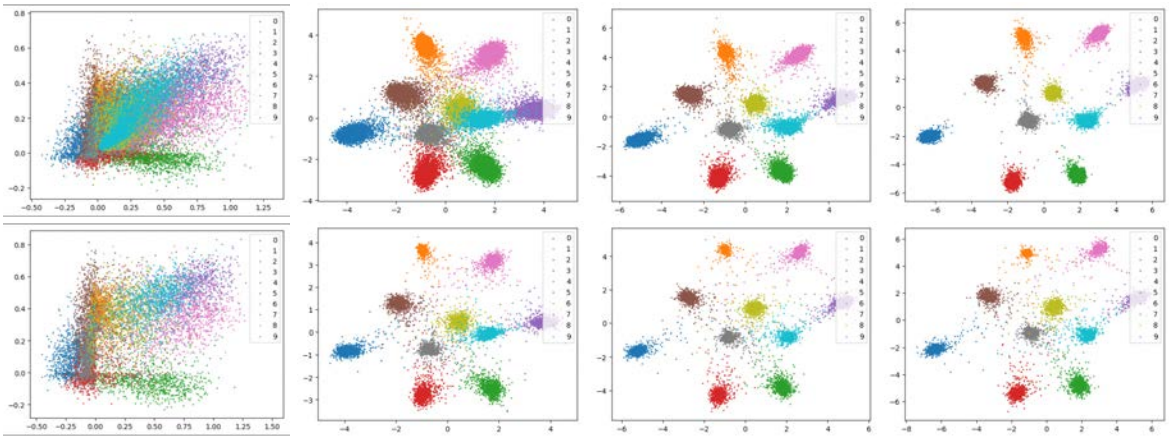


Figure 6.2 Visualizations of feature learning process on the training (first row) and testing set (second row), respectively.

representation is able to avoid overfitting.

Coupled with different base CNNs. We are also interested in how our DCSR-Net works in couple with different CNN architectures. We evaluate MLP, LISTA and DCSR-Net on VGG-16 and another popular yet deeper architecture ResNet-101 [61]. We use the *pool5* of ResNet-101 as the input to DCSR-Net. The results are shown in Table 6.2. One can see that DCSR-Net greatly outperforms MLP and LISTA with more than 3% improvement on DTD and 5% improvement on KTH-T2b, CUB and Aircraft. On the other hand, coupling DCSR-Net with ResNet-101 performs better than coupling with VGG-16. This is not a surprise since ResNet-101 is much deeper than VGG-16.

Comparison with structured loss functions. It is necessary to compare DCSR-Net with those structured loss functions, such as center loss [176] and angular softmax loss [96], under the same VGG-16 base network to explore the performance of learned feature representation. The center loss aims to minimize the distances between the image features (network output) and their class centers in the embedding space, while the angular softmax loss is indeed a large-margin loss function that attains the separated structure with good generalization. The results are shown in Table 6.3. One can observe that the center loss is comparable to angular softmax

Table 6.2 (Mean) Classification accuracy (%) with different CNN architectures and network baselines.

CNNs	Networks	DTD	KTH-T2b	CUB	Aircraft
VGG-16	MLP	68.3	72.3	79.2	79.5
	LISTA	68.5	71.1	78.2	78.7
	DCSR-Net	71.7	76.6	83.5	83.4
ResNet-101	MLP	71.8	73.6	80.7	81.9
	LISTA	69.7	72.4	79.5	82.1
	DCSR-Net	73.3	77.2	85.4	85.3

Table 6.3 (Mean) Classification accuracy (%) with different loss functions.

Loss	DTD	KTH-T2b	CUB	Aircraft
Center loss [176]	69.5	74.5	82.1	81.9
Angular softmax loss [96]	70.0	75.6	82.6	82.2
DCSR-Net	71.7	76.6	83.5	83.4

loss, while DCSR-Net consistently outperforms both structured loss functions in all the used datasets, validating the superior performance of DCSR-Net for learning discriminative feature representations from data.

6.5.2 Experiments on Texture Classification

Effectiveness of DCSR-Net. We consider another two embedding baselines under VGG-16 to verify the effectiveness of DCSR-Net: FC-CNN and Compact Bilinear Pooling (CBP) [42]. The former is the standard fine-tuning setting with three fully-connected (FC) layers. To effectively exploit the rich statistics of convolutional features, we can use CBP [42] as the pooling step instead of global average pooling on *relu5_3* layer, then pass the features into DCSR-Net. As shown in Table 6.4(a), by deploying DCSR-Net as the structured modelling,

Table 6.4 DCSR-Net obtains state-of-the-art performance on two texture classification datasets (a),(b) and two fine-grained classification datasets (c),(d). Improvement over the baseline model is reported as (Δ).

(a) DTD

Method	Accuracy	Δ
FV-CNN [25]	70.6 (±0.9)	-
BCNN [94]	71.5 (±0.8)	-
LFV [152]	72.7 (±1.0)	-
Deep-TEN [194]	72.9 (±0.9)	-
FC-CNN	68.3 (±1.2)	(3.4)
DCSR-CNN	71.7 (±1.1)	
CBP [42]	71.2 (±0.9)	(3.1)
DCSR-CBP	**74.3 (±1.0)**	

(b) KTH-T2b

Method	Accuracy	Δ
FV-CNN [25]	75.9 (±2.4)	-
BCNN [94]	76.4 (±3.5)	-
LFV [152]	77.1 (±3.1)	-
Deep-TEN [194]	78.5 (±3.3)	-
FC-CNN	72.3 (±3.4)	(4.3)
DCSR-CNN	76.6 (±3.0)	
CBP [42]	75.8 (±2.9)	(3.0)
DCSR-CBP	**78.8 (±3.1)**	

(c) CUB

Method	Accuracy	Δ
BCNN [94]	84.1	-
PDFS [201]	84.5	-
RA-CNN [39]	85.3	-
HIHCA [13]	85.3	-
FC-CNN	79.2	(4.3)
DCSR-CNN	83.5	
CBP [42]	83.7	(2.2)
DCSR-CBP	**85.9**	

(d) Aircraft

Method	Accuracy	Δ
Symbiotic [18]	72.5	-
FV-FGC [49]	80.7	-
BCNN [94]	84.1	-
HIHCA [13]	**88.3**	-
FC-CNN	79.5	(3.9)
DCSR-CNN	83.4	
CBP [42]	83.6	(3.7)
DCSR-CBP	87.3	

we obtain substantial improvements of 3.4% and 3.1% (on the DTD dataset) over FC-CNN and CBP, respectively. The similar phenomenon can be observed on the KTH-T2b dataset with the accuracy improvements of 4.3% and 3.0% (see Table 6.4(b)).

Comparison with state-of-the-arts. We compare DCSR-Net with the following state-of-the-art feature encoding based approaches: FV-CNN [25], BCNN [94], LFV [152] and Deep-TEN [194]. FV-CNN and LFV perform Fisher Vector (FV) encoding on a particular layer of the CNN, and the latter further designs locally-transferred FV descriptors via a multi-layer neural network. BCNN and Deep-TEN can be regarded as order-less pooling methods that integrate an encoding layer on the convolutional activations. By simply applying DCSR-Net on the last conv layer of VGG-16, our DCSR-CNN delivers highly competitive performance, possibly because it is designed to favor class-aware structure, which works for texture classification. Furthermore, our DCSR-CBP provides consistent improvements over the state-of-the-art Deep-TEN by achieving an average classification rate of 74.3% on DTD and 78.8% on KTH-T2b.

6.5.3 Experiments on Fine-grained Visual Categorization

Effectiveness of DCSR-Net. We use the same baselines as adopted in Section 6.5.2 for FGVC. As shown in Table 6.4(c) and Table 6.4(d), compared with FC-CNN and CBP baselines, the accuracy improvements are also promising: 4.3% and 2.2% on the CUB dataset, and 3.9% and 3.7% on the Aircraft dataset.

Comparison with state-of-the-arts. For fair comparison, we compare DCSR-Net with several state-of-the-art methods that use only image-level labels, including BCNN [94], PDFS [201], RA-CNN [39], HIHCA [13] on the CUB dataset, and Symbiotic [18], FV-FGC [49], BCNN [94], HIHCA [13] on the Aircraft dataset. On the CUB dataset, DCSR-CNN achieves a competitive accuracy of 83.5% compared to BCNN and PDFS. By using a stronger base network, DCSR-CBP reaches 85.9% and outperforms RA-CNN and HIHCA (85.3%). On

the Aircraft dataset, DCSR-CNN obtains a similar result to BCNN (83.4% vs 84.1%). The accuracy can be significantly improved by using CBP baseline (87.3%), which is competitive with HIHCA (88.3%). However, please note that HIHCA adopts a multi-layer fusion scheme while DCSR-CBP only uses the one-layer feature.

6.6 Conclusion

We proposed a structured and compact network, namely DCSR-Net, whose architecture was carefully designed by referring to a class-aware sparse model that learns discriminative centralized sparse representations with small intra-class variances. DCSR-Net provides a flexible and structured modelling built upon existing CNNs to make use of the merits of the structure insights of optimization-based methods, which are particularly helpful for recognition tasks with limited training data. Experiments on texture and fine-grained classification showed that DCSR-Net is very cost-effective for learning highly discriminative representations.

Chapter 7

Conclusion

7.1 Summary and Contributions

In this thesis, we explored classic sparse models and powerful deep learning tools (CNNs and RNNs) for better visual recognition. The discriminative models and representation learning approaches presented in this research enabled us to address some of the innate challenges in several vision tasks, and the main contributions can be summarized as follows:

We first provide a clear probabilistic interpretation for conventional representation based classifiers used in the era of shallow learning architectures such as SRC and CRC. We propose a probabilistic collaborative representation framework, where the probability that a test sample belongs to the collaborative subspace of all classes can be well defined and computed. Consequently, we present ProCRC to jointly maximize the likelihood that a test sample belongs to each of the multiple classes. The proposed ProCRC shows superior performance to many popular classifiers, including SRC, CRC and SVM. Coupled with the CNN features, it also leads to state-of-the-art classification results on a variety of challenging visual datasets.

We then investigate the rich statistics of CNN activations for FGVC. The success of FGVC extremely relies on the modeling of appearance and interactions of various semantic parts, which makes it very challenging because: (i) part annotation and detection require expert

guidance and are very expensive; (ii) parts are of different sizes; and (iii) the part interactions are complex and of higher-order. To address these issues, we propose an end-to-end framework based on the higher-order integration of hierarchical convolutional activations for FGVC. A polynomial kernel based predictor is proposed to capture higher-order statistics of convolutional activations for modelling part interaction. To model inter-layer part interactions, we extend polynomial predictor to integrate hierarchical activations via kernel fusion. The proposed framework yields more discriminative representation and achieves competitive results on the widely used FGVC datasets.

We also consider weakly-supervised learning of external data for video summarization. Video summarization is a challenging under-constrained problem because the underlying summary of a single video strongly depends on users' subjective understandings. To leverage the plentiful web-crawled videos to improve the performance of video summarization, we present a generative modelling framework VESD to learn the latent semantic video representations which act as a bridge between benchmark data and web data. Specifically, our VESD couples two important components: a variational autoencoder for learning the latent semantics from web videos, and an encoder-attention-decoder for saliency estimation of raw video and summary generation. A loss term to learn the semantic matching between the generated summaries and web videos is presented, and the overall framework is further formulated into a unified conditional variational encoder-decoder. Experiments conducted on the challenging and diverse summarization datasets demonstrate the superior performance of our approach to existing state-of-the-art methods

We further consider how to combine classic sparse models with DNNs for representation learning. Despite the remarkable success of deep architectures such as CNNs for image classification in recent years, it still remains a challenging task to learn highly discriminative representations from datasets of limited size, mainly due to the lack of cost-effective structured modelling of networks. Inspired by the merits of the trainable architecture of sparse models,

we propose a novel structured network, which could produce discriminative centralized sparse representations to exploit the discriminative structure of small intra-class variance. The so-called DCSR-Net implements a truncated module of sparse optimization and can be cascaded to existing deep architectures with negligible additional parameter complexity. Experiments demonstrate that coupling DCSR-Net with CNNs greatly facilitates the classification tasks with limited training data size.

7.2 Future Works

Visual recognition still lag far behind what human vision is capable of. The standard way of going about is to start from the task's goals and requirements, to design the proper models and efficient algorithms to learn good representations. In the future, we will expand our research in the following directions:

Learning prior knowledge. Prior assumptions are widely used by the proposed models in this thesis as they can help reduce data consumption and model complexity during learning and improve performance. However, just like the Gaussian prior adopted in both ProCRC and VESD, most of the prior knowledge is based on the human common sense or pre-defined setting but might not be suitable for different tasks or datasets. Therefore, adaptively learning prior knowledge from data or task at hand is an exciting direction to explore.

Using less supervision. In Chapters 4 and 5, we have proposed the cost-efficient frameworks for addressing FGVC and video summarization tasks. However, many real-world applications using DNNs still require extensive human annotations to obtain satisfactory performance, which motivates us to investigate effective models with less supervision, and combine more learning paradigms such as transfer learning, weakly-supervised learning, unsupervised learning and one/few-shot learning. This is inspired by the fact that humans do not use massive of annotations when they learn to understand the visual environment.

Studying interpretable models and representations. Although deep learning approaches

have achieved tremendous success in recent years, they are usually treated as black boxes and therefore less preferred in many applications where interpretation is needed. In this thesis, we manage to incorporate sophisticated sparse models and machine learning methods into DNNs, which is just a starting point for the interpretability goal. We believe that high model interpretability is of significant value in both theory and practice and studying interpretable models and representations is a prospective trend in the future.

Bibliography

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 2, 7

[2] Shivani Agarwal and Dan Roth. Learning a sparse representation for object detection. In *European Conference on Computer Vision*, pages 113–127, 2002. 2

[3] Michal Aharon, Michael Elad, and Alfred Bruckstein. *k*-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006. 4

[4] Anelia Angelova and Shenghuo Zhu. Efficient object detection and segmentation for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 811–818, 2013. 39

[5] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016. 6, 46, 88

[6] Artem Babenko and Victor Lempitsky. Aggregating local deep features for image retrieval. In *IEEE International Conference on Computer Vision*, pages 1269–1277, 2015. 47

[7] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*,

pages 153–160, 2007. 2

[8] Thomas Berg and Peter N Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 955–962, 2013. 38

[9] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Multipath sparse coding using hierarchical matching pursuit. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 660–667, 2013. 39

[10] Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 22, 23

[11] Steve Branson, Grant Van Horn, Serge Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014. 38, 43, 44

[12] Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *European Conference on Computer Vision*, pages 438–451, 2010. 44

[13] Sijia Cai, Wangmeng Zuo, and Lei Zhang. Higher-order integration of hierarchical convolutional activations for fine-grained visual categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–520, 2017. 97, 101, 102

[14] Sijia Cai, Wangmeng Zuo, Lei Zhang, Xiangchu Feng, and Ping Wang. Support vector guided dictionary learning. In *European Conference on Computer Vision*, pages 624–639, 2014. 3

[15] Barbara Caputo, Eric Hayman, and P Mallikarjuna. Class-specific material categorisation. In *IEEE International Conference on Computer Vision*, pages 1597–1604, 2005. 97

[16] Joao Carreira, Rui Caseiro, Jorge Batista, and Cristian Sminchisescu. Semantic seg-

mentation with second-order pooling. In *European Conference on Computer Vision*, pages 430–443, 2012. 49

[17] Yuning Chai, Victor Lempitsky, and Andrew Zisserman. Bicos: A bi-level co-segmentation method for image classification. In *IEEE International Conference on Computer Vision*, pages 2579–2586, 2011. 39

[18] Yuning Chai, Victor Lempitsky, and Andrew Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *IEEE International Conference on Computer Vision*, pages 321–328, 2013. 61, 65, 101, 102

[19] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014. 39

[20] Yuejie Chi and Fatih Porikli. Connecting the dots in multi-class classification: From nearest subspace to collaborative representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3602–3609, 2012. 24, 32

[21] Yuejie Chi and Fatih Porikli. Classification and boosting with multiple collaborative representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1519–1531, 2014. 24, 32

[22] Jen-Tzung Chien and Chia-Chen Wu. Discriminant waveletfaces and nearest feature classifiers for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1644–1649, 2002. 23, 32

[23] Wen-Sheng Chu, Yale Song, and Alejandro Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3584–3592, 2015. 68, 70, 78, 79, 82

[24] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014. 97

[25] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3828–3836, 2015. 6, 38, 46, 87, 88, 101, 102

[26] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, 2012. 46

[27] George E Dahl, Tara N Sainath, and Geoffrey E Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8609–8613, 2013. 2

[28] Weihong Deng, Jiani Hu, and Jun Guo. Extended src: Undersampled face recognition via intraclass variant dictionary. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1864–1870, 2012. 24

[29] Mandar Dixit, Si Chen, Dashan Gao, Nikhil Rasiwasia, and Nuno Vasconcelos. Scene classification with semantic fisher vectors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2015. 46

[30] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015. 67

[31] Matthias Dorfer, Rainer Kelz, and Gerhard Widmer. Deep linear discriminant analysis. *arXiv preprint arXiv:1511.04707*, 2015. 90

[32] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on Learning Theory*, pages 907–940, 2016. 2

[33] Ehsan Elhamifar, Guillermo Sapiro, and Rene Vidal. See all by looking at a few: Sparse modeling for finding representative objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1600–1607, 2012. 70, 82

[34] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008. 32

[35] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941, 2016. 5

[36] Jiashi Feng, Bingbing Ni, Qi Tian, and Shuicheng Yan. Geometric p-norm feature pooling for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2609–2704, 2011. 48

[37] Shikun Feng, Zhen Lei, Dong Yi, and Stan Z Li. Online content-aware video condensation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2082–2087, 2012. 70

[38] Vojtech Franc, Alexander Zien, and Bernhard Schölkopf. Support vector machines as probabilistic models. In *International Conference on Machine Learning*, pages 665–672, 2011. 23

[39] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 101, 102

[40] Shenghua Gao, Ivor Wai-Hung Tsang, Liang-Tien Chia, and Peilin Zhao. Local features are not lonely–laplacian sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3555–3561, 2010. 3

[41] Shenghua Gao, IW Tsang, and Yi Ma. Learning category-specific dictionary and shared dictionary for fine-grained image categorization. *IEEE Transactions on Image Processing*, 2013. 4

[42] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 317–326,

2016. 47, 54, 100, 101

[43] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015. 6

[44] Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001. 32, 34

[45] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. 44

[46] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*, pages 2069–2077, 2014. 70, 84

[47] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *European Conference on Computer Vision*, pages 392–407, 2014. 46

[48] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 71

[49] Philippe-Henri Gosselin, Naila Murray, Hervé Jégou, and Florent Perronnin. Revisiting the fisher vector for fine-grained classification. *Pattern Recognition Letters*, 49:92–98, 2014. 61, 65, 101, 102

[50] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013. xiv, 19, 20

[51] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005. 19

[52] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *International Conference on International Conference on Machine Learning*, pages 399–406, 2010. 88, 89, 94, 98

[53] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 32, 36, 37

[54] Genliang Guan, Zhiyong Wang, Shaohui Mei, Max Ott, Mingyi He, and David Dagan Feng. A top-down approach for video summarization. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 11(1):4, 2014. 70

[55] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *European Conference on Computer Vision*, pages 505–520, 2014. 70

[56] Michael Gygli, Helmut Grabner, and Luc Van Gool. Video summarization by learning submodular mixtures of objectives. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3090–3098, 2015. 70, 79, 84

[57] Michael Gygli, Yale Song, and Liangliang Cao. Video2gif: Automatic generation of animated gifs from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1001–1009, 2016. 67, 68, 70

[58] Jihun Ham, Daniel D Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *International Conference on Machine Learning*, page 47, 2004. 27

[59] Jihun Hamm and Daniel D Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *International Conference on Machine Learning*, pages 376–383, 2008. 23

[60] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 447–456, 2015. 46, 47, 53, 60

[61] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5, 55, 82, 87, 99

[62] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005. 27

[63] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. 2

[64] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 18

[65] http://yann.lecun.com/exdb/mnist/. The mnist database of handwritten digits, 2011. 32, 97

[66] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017. 5

[67] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015. 2

[68] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 60, 61

[69] Herve Jegou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, 34(9):1704–1716, 2012. 1

[70] Shihao Ji, Ya Xue, and Lawrence Carin. Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56(6):2346–2356, 2008. 4

[71] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013. 5

[72] Xudong Jiang and Jian Lai. Sparse and dense hybrid representation via dictionary decomposition for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(5):1067–1079, 2015. 24

[73] Zhuolin Jiang, Zhe Lin, and L Davis. Label consistent k-svd: learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. 4

[74] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Learning a discriminative dictionary for sparse coding via label consistent k-svd. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1697–1704, 2011. 4

[75] Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *International Conference on Machine Learning*, volume 22, pages 583–591, 2012. 50, 56

[76] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. 67

[77] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost van de Weijer, Andrew D Bagdanov, Antonio M Lopez, and Michael Felsberg. Coloring action recognition in still images. *International Journal of Computer Vision*, 105(3):205–221, 2013. 38

[78] Fahad Shahbaz Khan, Joost van de Weijer, Rao Muhammad Anwer, Michael Felsberg, and Carlo Gatta. Semantic pyramids for gender and action recognition. *IEEE Transactions on Image Processing*, 23(8):3633–3645, 2014. 38

[79] Gunhee Kim, Leonid Sigal, and Eric P Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. 2014. 68, 70, 82

[80] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 20, 68, 71

[81] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 50, 51

[82] Jonathan Krause, Hailin Jin, Jianchao Yang, and Li Fei-Fei. Fine-grained recognition without part annotations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5546–5555, 2015. 60, 61

[83] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013. 55

[84] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 5, 40

[85] Dmitry Laptev, Nikolay Savinov, Joachim M Buhmann, and Marc Pollefeys. Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 289–297, 2016. 6

[86] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015. 75

[87] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian

process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816, 2005. 23

[88] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, 2006. 2

[89] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *International Conference on Machine Learning*, 2013. 46

[90] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *International Conference on Machine Learning*, 2015. 46

[91] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning*, pages 609–616, 2009. 2

[92] Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C Weng. A note on platt's probabilistic outputs for support vector machines. *Machine Learning*, 68(3):267–276, 2007. 23

[93] Tsung-Yu Lin and Subhransu Maji. Visualizing and understanding deep texture representations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2791–2799, 2016. 88

[94] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *IEEE International Conference on Computer Vision*, pages 1449–1457, 2015. 6, 46, 50, 54, 56, 60, 61, 62, 65, 87, 88, 97, 101, 102

[95] Lingqiao Liu, Chunhua Shen, and Anton van den Hengel. The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4749–4757, 2015. 47

[96] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 90, 99, 100

[97] Yanan Liu, Fei Wu, Zhihua Zhang, Yueting Zhuang, and Shuicheng Yan. Sparse representation using nonnegative curds and whey. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3578–3585, 2010. 3

[98] Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863, 2014. 45

[99] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 47

[100] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial lstm networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 72, 76, 77, 80, 82, 83

[101] Julien Mairal, Francis Bach, and Jean Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012. 4

[102] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and et al. Supervised dictionary learning. In *Advances in Neural Information Processing Systems*, 2008. 4

[103] Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In *Advances in Neural Information Processing Systems*, pages 2627–2635, 2014. 45

[104] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 55, 97

[105] Aleix M Martinez. The ar face database. *CVC Technical Report*, 24, 1998. 32, 34

[106] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015. 72

[107] Baback Moghaddam. Principal manifolds and probabilistic subspaces for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):780–788, 2002. 23, 24

[108] Baback Moghaddam, Tony Jebara, and Alex Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, 2000. 23

[109] Baback Moghaddam and Alex Pentland. Probabilistic visual learning for object detection. In *IEEE International Conference on Computer Vision*, pages 786–793, 1995. 23, 24

[110] Baback Moghaddam and Alex Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710, 1997. 23, 24

[111] Mohammad Moghimi, Serge J Belongie, Mohammad J Saberian, Jian Yang, Nuno Vasconcelos, and Li-Jia Li. Boosted convolutional neural networks. In *British Machine Vision Conference*, 2016. 60, 61

[112] Arthur G Money and Harry Agius. Video summarisation: A conceptual framework and survey of the state of the art. *Journal of Visual Communication and Image Representation*, 19(2):121–143, 2008. 70

[113] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2924–2932, 2014. 2

[114] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012. 22

[115] Naila Murray and Florent Perronnin. Generalized max pooling. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2473–2480, 2014. 39, 48

[116] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, pages 807–814, 2010. 2

[117] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals,

Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015. 67

[118] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008. 32, 36, 37

[119] Rameswar Panda, Abir Das, Ziyan Wu, Jan Ernst, and Amit K Roy-Chowdhury. Weakly supervised summarization of web videos. In *IEEE International Conference on Computer Vision*, pages 3677–3686, 2017. 71, 79, 84

[120] Rameswar Panda and Amit K Roy-Chowdhury. Collaborative summarization of topic-related videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 68, 79, 82

[121] Rameswar Panda and Amit K Roy-Chowdhury. Sparse modeling for topic-oriented video summarization. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1388–1392, 2017. 70

[122] Florent Perronnin and Diane Larlus. Fisher vectors meet neural networks: A hybrid classification architecture. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3743–3752, 2015. 6

[123] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision*, pages 143–156, 2010. 1

[124] Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 239–247, 2013. 49

[125] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 10(3):61–74,

1999. 23

[126] Bryan A Plummer, Matthew Brown, and Svetlana Lazebnik. Enhancing video summarization via vision-language embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 68, 70

[127] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In *European Conference on Computer Vision*, pages 540–555, 2014. 72, 79, 84

[128] Simon JD Prince and James H Elder. Probabilistic linear discriminant analysis for inferences about identity. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007. 23

[129] Yael Pritch, Alex Rav-Acha, Avital Gutman, and Shmuel Peleg. Webcam synopsis: Peeking around the world. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007. 70

[130] Ignacio Ramírez, Federico Lecumberry, and Guillermo Sapiro. Universal priors for sparse modeling. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pages 197–200, 2009. 3

[131] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 4

[132] Ali S Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014. 6, 39

[133] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016. 72

[134] Yong Rui, Anoop Gupta, and Alex Acero. Automatically extracting highlights for tv

baseball programs. In *ACM International Conference on Multimedia*, pages 105–115, 2000. 71

[135] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 2, 7, 32, 40, 55, 87, 96

[136] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, 105(3):222–245, 2013. 1

[137] Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2774–2781, 2014. 89

[138] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015. 90

[139] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. 19

[140] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 5

[141] Fahad Shahbaz, Jiaolong Xu, Joost van de Weijer, Andrew Bagdanov, Muhammad Anwer Rao, and Antonio Lopez. Recognizing actions through action-specific person detection. *IEEE Transactions on Image Processing*, 24(11):4422–4432, 2015. 38

[142] Aidean Sharghi, Boqing Gong, and Mubarak Shah. Query-focused extractive video summarization. In *European Conference on Computer Vision*, pages 3–19, 2016. 70

[143] Gitika Sharma, Frédéric Jurie, and Cordelia Schmid. Expanded parts model for human attribute and action recognition in still images. In *IEEE Conference on Computer Vision*

and Pattern Recognition, pages 652–659, 2013. 38

[144] Marcel Simon and Erik Rodner. Neural activation constellations: Unsupervised part model discovery with convolutional networks. In *IEEE International Conference on Computer Vision*, pages 1143–1151, 2015. 38, 39, 43, 44, 54

[145] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 5, 46

[146] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5, 6, 37, 39, 54, 87, 96

[147] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, page 1470, 2003. 1

[148] Alan F Smeaton, Paul Over, and Wessel Kraaij. Evaluation campaigns and trecvid. In *ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, 2006. 78

[149] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4080–4090, 2017. 91

[150] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015. 76

[151] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5179–5187, 2015. 68, 78, 85

[152] Yang Song, Fan Zhang, Qing Li, Heng Huang, Lauren J ODonnell, and Weidong Cai.

Locally-transferred fisher vectors for texture classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4912–4920, 2017. 101, 102

[153] Pablo Sprechmann, Alexander M Bronstein, and Guillermo Sapiro. Learning efficient sparse and low rank models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1821–1833, 2015. 88, 94

[154] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 2

[155] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *Advances in Neural Information Processing Systems*, pages 10–18, 2016. 89

[156] Min Sun, Ali Farhadi, and Steve Seitz. Ranking domain-specific highlights by analyzing edited videos. In *European Conference on Computer Vision*, pages 787–802, 2014. 71

[157] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014. 73

[158] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 5, 55, 60, 80, 82

[159] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 5

[160] Hao Tang, Vivek Kwatra, Mehmet Emre Sargin, and Ullas Gargi. Detecting highlights in sports videos: Cricket as a test case. In *IEEE International Conference on Multimedia and Expo*, pages 1–6, 2011. 71

[161] Dacheng Tao, Xuelong Li, Weiming Hu, Stephen Maybank, and Xindong Wu. Supervised tensor learning. In *IEEE International Conference on Data Mining*, pages 8–pp, 2005. 51

[162] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999. 23

[163] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE International Conference on Computer Vision*, pages 4489–4497, 2015. 5, 82

[164] Ba Tu Truong and Svetha Venkatesh. Video abstraction: A systematic review and classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(1):3, 2007. 70

[165] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *ACM International Conference on Multimedia*, pages 1469–1472, 2010. 37

[166] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 72

[167] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 32, 36, 55, 97

[168] Catherine Wah, Grant Van Horn, Steve Branson, Subhransu Maji, Pietro Perona, and Serge Belongie. Similarity comparisons for interactive fine-grained categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–866, 2014. 44

[169] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference*

on Computer Vision, pages 835–851, 2016. 76

[170] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367. IEEE, 2010. 3, 24, 38, 39

[171] Ruiping Wang, Shiguang Shan, Xilin Chen, and Wen Gao. Manifold-manifold distance with application to face recognition based on image set. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 23

[172] Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Proximal deep structured models. In *Advances in Neural Information Processing Systems*, pages 865–873, 2016. 89

[173] Yaming Wang, Jonghyun Choi, Vlad Morariu, and Larry S Davis. Mining discriminative triplets of patches for fine-grained classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1163–1172, 2016. 44

[174] Zhangyang Wang, Qing Ling, and Thomas Huang. Learning deep l0 encoders. In *AAAI Conference on Artificial Intelligence*, pages 2194–2200, 2016. 89

[175] Zhaowen Wang, Jianchao Yang, Nasser Nasrabadi, and Thomas Huang. Look into sparse representation-based classification: a margin-based perspective. In *IEEE International Conference on Computer Vision*, 2013. 4

[176] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515, 2016. 90, 91, 99, 100

[177] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009. 2, 3, 13, 24, 30, 32, 88

[178] Ruobing Wu, Baoyuan Wang, Wenping Wang, and Yizhou Yu. Harvesting discriminative meta objects with deep cnn features for scene classification. In *IEEE International*

Conference on Computer Vision, pages 1287–1295, 2015. 6

[179] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *IEEE International Conference on Computer Vision*, pages 1395–1403, 2015. 46, 47, 60

[180] Bo Xin, Yizhou Wang, Wen Gao, David Wipf, and Baoyuan Wang. Maximal sparsity with deep networks? In *Advances in Neural Information Processing Systems*, pages 4340–4348, 2016. 89

[181] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015. 73

[182] Zhongwen Xu, Yi Yang, and Alex G Hauptmann. A discriminative cnn video representation for event detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1798–1807, 2015. 6, 46

[183] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2129–2137, 2016. 47

[184] Huan Yang, Baoyuan Wang, Stephen Lin, David Wipf, Minyi Guo, and Baining Guo. Unsupervised extraction of video highlights via robust recurrent auto-encoders. *arXiv preprint arXiv:1510.01442*, 2015. 71

[185] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009. 2, 3, 24, 38, 39, 88

[186] Junfeng Yang and Yin Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2011. 32, 34, 35

[187] Meng Yang, David Zhang, and Xiangchu Feng. Fisher discrimination dictionary learning for sparse representation. In *IEEE International Conference on Computer Vision*, 2011. 4

[188] Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *IEEE International Conference on Computer Vision*, pages 1476–1483, 2015. 46

[189] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *IEEE International Conference on Computer Vision*, pages 1331–1338, 2011. 32, 36, 38

[190] Ting Yao, Tao Mei, and Yong Rui. Highlight detection with pairwise deep ranking for first-person video summarization. 2016. 70

[191] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833, 2014. 39

[192] Haichao Zhang, Jianchao Yang, Yanning Zhang, Nasser M Nasrabadi, and Thomas S Huang. Close the loop: Joint blind image restoration and recognition with sparse representation prior. In *IEEE International Conference on Computer Vision*, pages 770–777, 2011. 24

[193] Han Zhang, Tao Xu, Mohamed Elhoseiny, Xiaolei Huang, Shaoting Zhang, Ahmed Elgammal, and Dimitris Metaxas. Spda-cnn: Unifying semantic part detection and abstraction for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1143–1152, 2016. 43, 44, 60, 61, 88

[194] Hang Zhang, Jia Xue, and Kristin Dana. Deep ten: Texture encoding network. *arXiv preprint arXiv:1612.02844*, 2016. 101, 102

[195] Hao Zhang, Alexander C Berg, Michael Maire, and Jitendra Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *IEEE*

Conference on Computer Vision and Pattern Recognition, volume 2, pages 2126–2136, 2006. 23

[196] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Summary transfer: Exemplar-based subset selection for video summarization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1059–1067, 2016. 70

[197] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *European Conference on Computer Vision*, pages 766–782, 2016. 67, 70, 72, 73, 79, 80, 84

[198] Lei Zhang, Meng Yang, and Xiangchu Feng. Sparse representation or collaborative representation: Which helps face recognition? In *IEEE International Conference on Computer Vision*, pages 471–478, 2011. 3, 13, 24, 27, 32

[199] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European Conference on Computer Vision*, pages 834–849, 2014. 43, 44, 60, 61, 88

[200] Qiang Zhang and Baoxin Li. Discriminative k-svd for dictionary learning in face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 4

[201] Xiaopeng Zhang, Hongkai Xiong, Wengang Zhou, Weiyao Lin, and Qi Tian. Picking deep filter responses for fine-grained image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1134–1142, 2016. 43, 44, 54, 60, 61, 101, 102

[202] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *IEEE International Conference on Computer Vision*, pages 1529–1537, 2015. 89

[203] Xin Zheng, Deng Cai, Xiaofei He, Wei-Ying Ma, and Xueyin Lin. Locality preserving clustering for image database. In *ACM International Conference on Multimedia*, pages

885–891, 2004. 27

[204] Yipin Zhou and Tamara L Berg. Learning temporal transformations from time-lapse videos. In *European Conference on Computer Vision*, pages 262–277, 2016. 76