UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering and Information Technology

Learning Robust Features for Recognition of Emotions in Images and Videos

by

Haimin Zhang

A THESIS SUBMITTED
IN FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

Sydney, Australia

2019

Certificate of Original Authorship

I, Haimin Zhang declare that this thesis, is submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy, in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. This document has not been submitted for qualifications at any other academic institution. This research is supported by the Australian Government Research Training Program.

Production Note:
Signature: Signature removed prior to publication.

Date: 20 November 2019

Learning Robust Features for Recognition of Emotions in Images and Videos

by

Haimin Zhang

Abstract

Today, recognition of emotions in images and videos has attracted increasing research attention. In terms of video emotion recognition, most existing approaches are based on spatial features extracted from video frames. The performance of these approaches is mainly restricted due to the broad affective gap between spatial image features and high-level emotions. To bridge the affective gap, we propose to recognize emotions with kernelized features. A polynomial kernel function is constructed based on rewritten the equation of the discrete Fourier transform as the linear kernel. Moreover, we propose to apply the sparse representation method to kernelized features to reduce the impact of noise contained in video frames. This method can further help contribute to performance improvement.

In the second work, we develop a weighted sum pooling method for video emotion representation. We present an end-to-end deep network for simultaneously image emotion classification and emotion intensity map prediction. The proposed network is build based on the feature pyramid network. The class activation mapping technique is utilized to generate pseudo intensity maps to train the network. The proposed network is first trained on a large-scale image emotion dataset and then used to extracted features and intensity maps for video frames. We empirically show that this approach is effective to improve recognition performance.

Recent work has shown that using local region information helps to improve image emotion recognition performance. In the third work, we develop an end-to-end deep neural network for image emotion recognition by utilizing emotion intensity. The proposed network is composed of an intensity prediction stream and a classification stream. The

class activation mapping technique is used to generated pseudo intensity maps to guide the intensity prediction network for emotion intensity learning. The predicted intensity maps are integrated to the classification stream for final recognition. The two streams are trained cooperatively with each other to improve the overall performance.

In the fourth work, we present a dual pattern learning network architecture with adversarial adaptation (DPLAANet). Unlike conventional networks, the proposed architecture has two input branches. The dual input structure allows the network to have a considerably large number of image pairs for training. This can help address the overfitting issue due to limited training data. Moreover, we introduce to use the adversarial training approach to reduce the domain difference between training data and test data. The experimental results show that the DPLAANets are effective for several benchmark datasets.

Thesis Supervisor: A/Prof. Min Xu
School of Electrical and Data Engineering

Acknowledgements

The four-year Ph.D study at UTS has been a wonderful experience. I would like to acknowledge several people who not only made this thesis well finished, but also brought a lot of support, joy, and happiness into this amazing academic journey.

First and foremost, I would like to sincerely thank my supervisor, A/Prof. Min Xu, for her continually supervision and encouragement during my Ph.D. study. We have collaborated on a number of awesome projects since I started the Ph.D program four years ago. Her sharp intuition and passion for knowledge have influenced me to dig deeper into the problems we are having and to discover something novel. This would be much helpful for my future career. I feel I am very fortunate to have been supervised by and working with her in the past four years.

I would like to express my thanks and appreciation to my co-supervisor Dr. Xiaoying Kong for help guidance and help. Many thanks to A/Prof. Qiang Wu and Dr. Wenjing Jia for their valuable advices and suggestions for my candidature assessment one and two, which are much helpful for improving the quality of this thesis. I would like to give thanks to A/Prof. Richard Xu, who has provided useful insight for my research and career. Special thanks go to Prof. Yu-gang Jiang at Fudan University who provided the datasets for conducting experiments.

I would like to acknowledge my labmates in the Aural and Visual Intelligence Lab: Dr. Tianrong Rao, Madhumita Takalkar, Lingxiang Wu, Zhongqin Wang, Zhiyuan Shi, Lei Sang, Yukun Yang, Wanneng Wu, Ruiheng Zhang, and Xiaoxu Li. I would like to thank my friends for their assistance: Dr. Cheng Luo and Dr. Ming Liu, Dr. Hao Li, Dr. Lin Ye, Dr. Zhichao Sheng, Dr. Ye Shi, etc.

I would like to acknowledge with gratitude the love of my family. Their support has always been unconditional. This thesis could not have been well finished without their

support.

Finally, I would like to thank the anonymous reviewers for reviewing this thesis.

List of Publications

The contents of this thesis are based on the following papers that have been published or accepted, or preprints that have been under submission or submitted to peer-reviewed journals.

Publications:

1. Haimin Zhang and Min Xu, "Recognition of Emotions in User-Generated Videos With Kernelized Features," *IEEE Transactions on Mulitmedia*, vol. 20, no. 10, pp. 2824-2835, 2018.

2. Haimin Zhang and Min Xu, "Modeling temporal information using discrete fourier transform for recognizing emotions in user-generated videos," *IEEE International Conference on Image Processing (ICIP)*, 2016.

3. Madhumita A. Takalkar, Haimin Zhang, and Min Xu, "Improving Micro-expression Recognition Accuracy Using Twofold Feature Extraction," *International Conference on MultiMedia Modeling (MMM)*, 2019.

4. Tianrong Rao, Xiaoxu Li, Haimin Zhang, and MinXu, "Multi-level region-based Convolutional Neural Network for image emotion classification," *Neurocomputing*, vol. 333, pp. 429-439, March, 2019.

5. Shenghong Hu, Min Xu, Haimin Zhang, Chunxia Xiao, and Chao Gui, "Affective Content-aware Adaptation Scheme on QoE Optimization of Adaptive Streaming over HTTP," accepted to *ACM Transactions on Multimedia Computing, Communications, and Applications* .

Others:

1. Haimin Zhang and Min Xu, "Weakly Supervised Emotion Intensity Prediction for Recognition of Emotions in Images," under review by *IEEE Transactions on Mulitmedia*.

2. Haimin Zhang and Min Xu, "Improving the Performance of Deep Networks by Dual Pattern Learning with Adversarial Adaptation," under first revision by *IEEE Transactions on Circuits and Systems for Video Technology*.

3. Haimin Zhang and Min Xu, "Frame-level Emotion Intensity Prediction for Improving Video Emotion Recognition Performance," under submission to *IEEE Transactions on Affective Computing*.

Contents

List of Figures

List of Tables

Abbreviation

- CNN: convolutional neural network
- RNN: recurrent neural networks
- SVM: support vector machine
- GAN: generative adversarial network
- DFT: discrete Fourier transform
- FFT: Fast Fourier transform
- FV: Fisher vector
- VLAD: vector of locality aggregated vectors
- CAM: class activation mapping
- RMSE: root mean square error
- RMSEL: Root mean square error in log space
- SGD: stochastic gradient descent
- ITE: image transfer encoding
- LLC: locality-constrained linear coding
- DPL: dual pattern learning
- ERM: empirical risk minimization
- HMM: hidden Markov model
- MFCC: Mel-frequency cepstral coefficients

- STE: short-time energy
- FPN: feature pyramid network
- CAN: collaborative and adversarial networks
- ADDA: adversarial discriminative domain adaptation
- SymNets: domain-symmetric networks
- LSTM: long short-term memory

Chapter 1

Introduction

1.1 Background and Problem Statement

Emotions play a significant role in peoples' lives. Psychological research has shown that human emotions can be evoked by visual stimuli such as images [1, 2] . While a lot of research efforts have been devoted to human facial expression recognition [3, 4], little progress has been made regarding recognition of emotions in images and videos. With the increasing popularity of social media platforms, huge amounts of images and video clips are uploaded to the Internet on a daily basis. This highlights the importance of developing intelligent algorithms to automatically predict the emotions that could be convoked by these images and videos. Visual emotion recognition has various potential applications, such image/video retrieval [5], recommendation, and opinion mining [6]. Over the years, visual emotion recognition has attracted increasing research attention in both the research community and industry [7, 8].

The aim of visual emotion recognition is to predict the invoked emotion in a person after they see a visual content such as an image or a video clip. It is also referred to as affective content analysis in some literature [9]. Unlike semantic recognition tasks such as image classification and scene classification, visual emotions have high intra-class variations. For instance, the images that can invoke the emotion of happy can be taken in vary different scenes. The high intra-class variations make visual emotion recognition a complicated task. Due to culture background difference or other reasons, an image may invoke different emotions in different people. Most existing benchmark datasets are labeled with the majority voting method, *i.e.,* the emotion that most voters selected is used as the emotion label. As a consequence, some similar images that contain the same emotion content are actually annotated with different emotions. This has been another challenge for visual emotion recognition. Today, deep neural networks (DNNs) have

become a dominant approach in various semantic recognition tasks due to their superior performance. Trained on a large-scale dataset, state-of-the-art deep neural networks have achieved human-level performance. However, the performance is not promising when they are directly applied for visual emotion recognition tasks. This has motivated us to develop efficient network architectures to improve emotion recognition performance.

Due to the limited size of existing video datasets for emotion recognition, deep learning models that have been developed for video classification and representation such as long short-term memory (LSTM) networks can not be directly applied for emotion recognition in videos. Existing approaches mainly focus on designing robust features at frame-level. Extracted frame-level features are aggregated by a pooling method such as average pooling to generate video-level features. Unlike emotion recognition in images, a video clip contains a number of video frames which are grouped together to convey a dominant emotion. Some frames in the video clip may convey a neutral emotion or even another emotion other than the dominant emotion. Modelling the dominant emotion information from all video frames is a challenging problem. In addition, video clips may contain high levels of noise, such as image blur introduced by camera moving and low-resolution of economical cameras. This further makes it difficult for emotion inference. In our first work, we tackle this problem by transforming frame-level spatial features to another space with a kernel function. We believe that the features in the new space could be more discriminative than original features. We developed an approach which is based on Fourier analysis for space transformation. Moreover, we applied the denoising method method to transformed features to reduce the impact of noise contained in videos. In our second work, we used an off-the-shelf CNN for frame-level intensity prediction and feature extraction. We empirically show that utilizing frame-level intensity helps to improve recognition accuracy.

With regard to emotion recognition in images, recent studies have shown that discovering discriminative image regions is effective to improve recognition performance. This is because emotion information is usually in a region within the whole image. Intuitively, emotion intensity maps could provide more detailed information than image regions. However, existing datasets are annotated with only image-level labels. Manually annotat-

ing emotion intensity maps would require huge amounts of work. There have been work showing that saliency regions can be derived from trained CNNs [10, 11]. Motivated by this method, we introduce a weakly supervised end-to-end network that simultaneously predicts emotion intensity maps and classification results. We show that utilizing predicted intensity maps help to improve recognition performance.

State-of-the-art deep neural networks usually contain many layers with a large number of parameters. This makes deep networks prone to overfit on training data. Training deep networks requires huge amounts of data to improve generalization performance. However, collecting data and annotating them require laborious work, especially when domain experts are necessary to distinguish between fine-grained visual categories. For some tasks, it is extremely difficult to collect samples. In our fourth work, we focus on efficient feature learning using CNNs with limited data. Inspired by the intuition that humans can learn knowledge from two images by analyzing and comparing them, we propose a network that is trained with image pairs. Our results demonstrate the effectiveness of the proposed network.

1.2 Thesis Objectives and Contributions

The objectives and main contributions of this thesis can be summarized as follows:

- We address the high intra-class variation problem and the noise issue in emotion recognition in videos. To this end, we propose a method to transform frame-level spatial features to another space with a kernel function. Kernelized features are demonstrated to be discriminative for video representation compared to spatial features. Moreover, we introduce to use the sparse representation method to smooth kernelized features to reduce the impact of noise in videos. We show that this method is effective to further improve recognition performance.

- To deal with the issue that different frames may invoke different levels of emotion. We propose to associate each frame with an emotion intensity value, which is predicted using a pre-trained deep network. The weighted sum pooling method is utilized for video-level feature generation. The experimental results demonstrate that

employing frame-level emotion intensity is helpful for performance improvement.

- Our third work focuses on employing emotion intensity to improve image emotion recognition performance. To this end, we propose a weakly supervised end-to-end network. The proposed network consists of two streams: an intensity prediction stream and a classification stream. The two steams cooperatively work with each other to improve overall performance.

- Our fourth work addresses the difficult that training deep networks requires huge amounts of data. We propose a network architecture that is trained with image pairs. In addition, we introduce to use the adversarial training approach to reduce the domain difference between training data and test data. The empirical study shows that the proposed network helps to improve the generalization performance, especially when training data are limited.

1.3 Thesis Outline

The rest of this thesis is organized as follows.

- In chapter 2, we briefly review related work on emotion modelling, deep neural networks, domain adaptation, kernel methods, and recognition of emotions in images and videos.

- Chapter 3 presents the proposed method for feature transformation with the kernel function which is derived based on Fourier analysis. The sparse representation method for smoothing transformed features is also introduced in this chapter.

- Chapter 4 presents the weighted sum pooling method for video representation. We introduce the network architecture that used for frame-level feature extraction and intensity approximation.

- Chapter 5 presents the proposed weakly supervised end-to-end network for emotion recognition in images. We introduce the technique to generate pseudo intensity maps, which are used to train the proposed network.

- In Chapter 6, we present dual pattern learning networks with adversarial adaptation.
- Chapter 7 summarizes this thesis and discusses potential directions in our future work.

Chapter 2

Related Work

Recognition of visual emotions is an important area of artificial intelligence. It has been researched for many years. In this chapter, we briefly review some related work that is relevant to the topic.

2.1 Emotion Modelling

In existing work on visual emotion recognition, emotion modelling is mainly based on the following two approaches: the categorical approach and the dimensional approach. In the categorical approach, emotions are represented by a number of discrete emotion classes. According to Ekman's [12] research based on studying the isolated culture of people in Papua New Guinea in 1972, there are six basic emotions, *i.e.*, happiness, sadness, surprise, fear, anger, and disgust, that can be invoked in the subjects by still images. Afterwards, many researches have confirmed that these six emotions are universal for all human beings. Psychologist Plutchik [13] developed a wheel representation method for eight basic emotions: joy, trust, fear, surprise, sadness, anticipation, anger, and disgust. The Plutchik's wheel of emotions illustrates various relationships among the eight emotions, including which ones are opposite to each other and which ones are close to each one. Unlike Plutchik's wheel of emotions, the Parrott's wheel of emotions [14] organizes emotions in a hierarchical structure with three levels. At the first level, emotions are categorized into two basic classes: positive and negative. There are six emotions at the second level, *i.e.,* anger, fear, joy, love, sadness, and surprise, and 25 fine-grained emotions at the third level.

In contrast to the categorical approach, emotions are annotated in a continuous space in the dimensional approach. The dimensional approach defines emotions with two/three fundamental dimensions along which the entire range of human emotions are distribut-

ed. This approach suggests that a common and interconnected neurophysiological system is responsible for all affective states [15]. The most commonly used spaces involve the 2-D valence-arousal space and the 3-D valence-arousal-control space [16]. Valence is a subjective feeling of pleasantness or unpleasantness; arousal is a subjective feeling of activated or deactivated [17]; and dominance represents the degree of control exerted by a stimulus ranging from in control to out of control [18]. Other dimensional approaches include the 3-D natural-temporal-energetic connotative space [19], the 3-D activity-weight-heat emotion factors [20].

The relationship between the two approaches are studied in [21]. Compared to the dimensional approach, the categorical approach is easy to understand and convenient for annotating emotions, and thus has been used in most studies. In this thesis, the categorical approach is adopted to model emotions in visual emotion recognition tasks.

2.2 Deep Neural Networks

Deep neural networks (DNNs) are a specific type of artificial neural networks. DNNs consist of a number of layers and can model very complex non-linear relationships between inputs and outputs. In recent years, DNNs have been seen tremendous growth in popularity in fields including computer vision, speech recognition, natural language processing, and machine translation. CNNs and recurrent neural networks (RNNs) are two of the most widely used types of DNNs.

CNNs have a long history in computer vision. Originally developed by LeCun *et al.* [22] in late 1980s, CNNs showed successful results for handwritten digit recognition. There are three basic ideas, *i.e.,* local receptive fields, shared weights, and pooling, in the architecture of CNNs. The idea of local receptive fields is inspired by biological processes [23]. In biological processes, individual cortical neurons respond to stimuli only in a restricted region of the visual field, which known as receptive field. Modern CNNs are usually vey deep in terms of depth with a large number of paraments. The success of modern CNNs has been achieved thanks to the availability of large-scale image datasets, such as ImageNet [24], and powerful GPU computing resources.

In 2012, Krizhevsky *et al.* [25] developed the AlexNet for ImageNet classification.

The architecture of AlexNet contains eight layers: five convolutional layers and three fully connected layers. This is the first modern CNN architecture proposed for large scale image classification. The AlexNet achieved considerably better performance than the previous state-of-the-art approaches. Many studies have been focused on designing CNN architectures to improve classification performance since the AlexNet. These studies include exploring increasing the depth (the number of layer) and increasing the width (the number of channels in each layer) of CNNs. For example, Simonyan *et al.* [26] investigated the impact of the network depth on its accuracy, and proposed VGGNets with 16 and 19 layers. In [27], He *et al.* introduced identity shortcut connections, and proposed the ResNet architecture. This architecture makes very deep networks easy to optimize. ResNets have achieved significant performance improvement for many tasks. In [28], Huang *et al.* explored increasing the width of networks and proposed densely connected networks (DenseNets). For each layer in DenseNets, the feature maps of all preceding layers are used as inputs. The DenseNet architecture encourages feature reuse, and can considerably reduce the number of parameters. In addition to exploring increasing depth and width, Xie *et al.* [29] researched increasing the cardinality of CNNs, which refers to the size of the set of transformations, and proposed the ResNeXt architecture. The ResNeXt is constructed by repeating a building block that aggregates a set of transformations with the same topology. They showed that increasing cardinality is effective to improve performance compared to increasing the depth or the width of CNNs.

Deep neural networks usually contain a large number of parameters; therefore, training deep neural networks requires huge amounts of data to reduce the overfitting issue. Enlarging training data by applying label-preserving transformations [25] can further help to reduce overfitting. Commonly used data augmentation methods include random crop, color jittering, horizontal or vertical flip of images. Recently, researcher started to use generative adversarial networks to generate adversarial samples for data augmentation. For example, Zheng *et al.* [30] proposed to use adversarial examples to improve person re-identification performance. Xie *et al.* [31] proposed to employ adversarial samples for semantic segmentation and object detection.

In addition to network development and data augmentation methods, researchers have

applied regularization techniques to improve the generalization performance. For example, Srivastava *et al.* [32] proposed a method referred to as dropout. The key idea of dropout is to randomly drop units of the neural network during training phase. The neurons which are dropped out in this way do not contribute to the forward pass and do not participate in back-propagation. This technique forces networks to learn more robust features that are useful in conjunction with many different random subsets of the other neurons. Ioffe *et al.* [33] proposed the batch normalization method. This batch normalization draws its strength from making normalization a part of the model architecture and performing the normalization for each training mini-batch. With the batch normalization method, we can use a higher learning rate and be less careful about parameters initialization to train deep neural networks.

More recently, researchers started to research on neural architecture search (NAS) to find a good network architecture. Most existing NAS approaches utilize evolutionary algorithms or reinforcement learning to automatically design network architectures. By now, NAS methods have outperformed manually designed architectures on some tasks such as image classification [34] and object detection. NAS approaches usually require extensive computing resources, which is not feasible for small research groups. For example, Zoph [34] *et al.* trained the NASNet on 500 GPUs for over four days to learn a good network architecture.

2.3 Domain Adaptation

Domain adaptation methods attempt to transfer the knowledge obtained from source domain to target domain. Usually, a large number annotated samples in source domains are utilized to facilitate learning in a new target domain. Domain adaptation is critical for success in new and unseen environments. Most existing studies can be summarized into two categories: (1) instance re-weighting, in which the samples from source domain are reused according to a weighting technique [35, 36], and (2) feature matching, which performs distribution alignment or subspace learning by exploiting the subspace geometrical structure to reduce the marginal or conditional distribution divergence between the source domain and the target domain.

Domain adaptation techniques have seen recent success through the merger with deep CNNs. Hoffman *et al.* [37] showed that when training data in the target domain is severely limited or unavailable, domain adaptation techniques as applied to CNNs can be more effective than the standard practice of fine-tuning. Over the last few years, the adversarial learning method has been researched for domain adaptation [38,39]. Adversarial learning approaches are based on generative adversarial networks (GANs) [40]. A typical GAN framework contains two models: a generative model G that captures the data distribution, and a discriminative model D that learns to determine whether a sample is from the training data or the generator G. The advantage of GAN over other generative methods is that there is no need for complex sampling or inference during training.

In [39], Tzeng *et al.* proposed the adversarial discriminative domain adaptation (ADDA) framework, which attempts to learn a discriminative mapping of target images to the source feature space by fooling a domain discriminator. The ADDA method achieved good performance on cross-domain digit classification and cross-modality object classification. In [41], Zhang proposed collaborative and adversarial networks (CAN) for unsupervised domain adaptation. In the CAN model, each CNN block is connected to a domain classifier. CAN models can learn domain informative representations at lower blocks by collaborative learning and learn domain uninformative representations from higher blocks by adversarial learning. Zhang *et al.* proposed domain-symmetric networks (SymNets) for unsupervised domain adaptation [42]. SymNets can address the limitation in aligning the joint distributions of feature and category across domains by using two-level domain confusion losses.

2.4 Kernel Methods

Kernel methods play an import role in machine learning and non-parametric statistics. The main idea of kernel methods is to map a vector $\boldsymbol{x}$ from the *input space* $\mathcal{X}$ to the *feature space* $\mathcal{F}$. The feature space could be of indefinite dimension. The mapping is performed with a mapping function ϕ:

$$\begin{aligned} \phi : \mathcal{X} &\to \mathcal{F} \\ \boldsymbol{x} &\mapsto \phi(x). \end{aligned} \tag{2.1}$$

For $\boldsymbol{x}$ and $\boldsymbol{x'}$ in the input space $\mathcal{X}$. A kernel function or a kernel k is represented as an inner product in a space $\mathcal{V}$:

$$k(\boldsymbol{x}, \boldsymbol{x'}) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}. \tag{2.2}$$

The word "kernel" is originally used in mathematics to denote a weighting function for a weighted sum or integral. If a problem can be represented in terms of inner production, the inner production can be replaced by a kernel function:

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x'}) \rangle_{\mathcal{V}} . \tag{2.3}$$

This is called the kernel trick. The kernel trick avoids the mapping function to be explicitly written out, as long as $\mathcal{V}$ is an inner product space. According to Mercer's theorem: an implicitly defined function ϕ exists whenever the space $\mathcal{X}$ is associated with a measure that ensures the function k satisfies Mercer's condition. Mercer's condition is similar to the definition of a positive-semidefinite matrix. A real-valued function satisfies Mercer's condition if the following equation holds for all square-integrable functions $g(x)$:

$$\int \int g(x)k(x, y)g(y)dxdy \geq 0. \tag{2.4}$$

Mercer's theorem provides the necessary and sufficient condition to guarantee a function to be a kernel function. Empirically, a function k that does not satisfy the Mercer's condition could perform reasonably if k is used to approximate the intuitive idea of similarity.

Commonly used kernels include polynomial kernel, radial basis function kernel, and sigmoid kernel. Kernel methods have been widely applied in many machine learning tasks. For instance, they play an essential role in classification problems (*e.g.,* support vector machine, Gaussian process classification), in regression problems (*e.g.,* kernel ridge regression, Gaussian process regression), and in clustering problems (*e.g.,* kernel k-Means, spectral clustering).

2.5 Emotion Recognition in Videos

Early studies on emotion recognition in videos were conducted on movie clips. In [43], Kang *et al.* investigated the relationship between emotional events and low-level features, and proposed to use the hidden Markov model (HMM) on motion, color and shot cut rate to detect emotional events in videos. Rasheed *et al.* [44] proposed a framework to perform high-level classification of previews into genres using low-level features, *i.e.*, average shot length, color variance, motion content and lighting key. In [45], Wang *et al.* proposed a systematic approach using features that are extracted based on psychology and cinematography for affective analysis in films. They also demonstrated that combining visual cues with cinematic principles are effective for genre categorization. In [46], Baveye *et al.* constructed a large video dataset which contains approximately 9,800 movie clips. They used an ensemble of SVM classifiers, which were trained on low-level visual and audio features such as zero-crossing rate and color harmony, for emotion recognition.

Instead of using only visual feature, many studies relied on combining multimodal features to improve recognition accuracy. This intuitively makes sense because humans perceive emotions from multimodalities including visual and auditory modalities. In the studies of [45] and [47], the authors investigated combining audio features, such as short-time energy (STE) and Mel-frequency cepstral coefficients (MFCC), with visual and for emotion recognition. The experimental results showed that audio features are complementary to visual features for recognition performance improvement.

Later, researchers started to research emotion recognition on web videos that are uploaded by Internet users. Compared to movie data, user-generated videos usually have much diverse content and be of low quality. Jiang *et al.* [48] collected a dataset from social media platforms, *i.e.,* YouTube and Flickr, which contains 1,101 videos. These videos were annotated with eight emotion categories according to Plutchiks wheel of emotions. This dataset has been a benchmark dataset for emotion recognition in videos. They evaluated a diverse of low-level visual and audio features for emotion recognition. In addition, they evaluated three types of attribute features, *i.e.*, Sentibank, ObjectBank [49], and Classemes [50]. Their experimental results demonstrated that attribute features are complementary to low-level features to improve recognition performance. In [5], an un-

supervised model was proposed on top of multimodal inputs, including visual, audio, and textual modalities, for the recognition of emotions in videos. The experimental results showed that the learned joint representations are complementary to hand-crafted features. The disadvantage of this approach is that using learned representations alone could not improve recognition accuracy. Moreover, a large auxiliary image and video dataset is required to train the model.

In contrast to hand-crafted features, deep features have shown to be much discriminative. Extracted from the activations from a CNN which is pre-trained on a large image dataset, deep features have achieved state-of-the-art performance on various visual recognition tasks. In the work of Chen *et al.* [51], event, object, and scene scores were extracted from state-of-the-art deep neural networks as features to represent the context information in videos. The extracted features were further integrated by a context fusion network to generate a unified representation for emotion recognition. The experimental results demonstrated that the proposed framework are effective to improve recognition performance. In the work of Xu *et al.* [52], deep features extracted from four types of CNNs, *i.e.,* AlexNet, VGGNet-16, VGGNet-19, and GoogleNet [53], were evaluated for emotion recognition. Moreover, the authors reported that fine-tuning CNNs on video frames or auxiliary image dataset could not improve recognition accuracy. In [54], Xu *et al.* proposed a technique to transfer knowledge from heterogeneous external sources including image and textual data, for understanding emotions in videos. They showed that their method are effective for three tasks, *i.e.,* video emotion recognition, attribution, and summarization. In [55], Li *et al.* proposed the deep decoupled video and "danmu" neural network jointly utilizing video content and user-generated texts for emotion analysis. More recently, Xu *et al.* [56] proposed a modality fusion framework which combines concept and content features extracted from action, scene, and object models for emotion recognition. They demonstrated that the concept and content features are effective to improve recognition accuracy.

2.6 Emotion Recognition in Images

Researchers started to research emotion recognition in images on small datasets. These datasets include the international affective picture system (IAPS) dataset [57] containing 1,182 documentary-style natural images, the ArtPhoto dataset [58] contanining 806 artistic photographs collected from a photo sharing site, and the Geneva affective picture database (GAPED) containing 520 negative images, 121 positive images, and 89 neutral images. Recently, large-scale datasets that are collected from social media platforms or searched from the Internet have emerged. These datasets include the FI dataset [59] collected from Flicker and Instagram and labeled by Amazon Mechanical Turk (AMT) workers, and the WEBEmo dataset which contains 268,000 images and is currently the largest dataset for images emotion recognition.

From the perspective of features, existing studies use two types of features: hand-crafted features and deep features. Hand-crafted features include low-level features, mid-level features, and high-level features. Low-level and mid-level image features are mainly extracted based on psychology study and art theory to model the emotional content of an image. Lu *et al.* [60] investigated the impact of shape features on emotion recognition by modeling roundness-angularity and simplicity-complexity. In [57], Machajdik *et al.* evaluated a number of low-level features such as color, texture, and harmonious composition for emotion recognition in images. In [61], Rao *et al.* proposed a multi-scale blocks based method using SIFT features as basic features for emotion recognition in images. In [62], Zhao *et al.* proposed to extract features based on principle-of-art, including balance, emphasis, harmony, variety, gradation, and movement, for image emotion classification.

High-level features can represent semantic concepts in images. They are more interpretable by humans and usually exhibit better performance than low- and mid-level features. Yuan *et al.* proposed the Sentriute method for image sentiment analysis [63]. In the Sentriute method, an image is represented as a response map of 102 predefined attribute features. In [64], Borth *et al.* proposed the SentiBank method, which consists of 1,200 semantic concept classifiers, to detect sentiments and emotions in visual content.

Unlike hand-crafted features which are extracted with algorithms using the informa-

tion present in the image, deep features are directly learned from image data. Deep features are much more robust compared to hand-crafted features and usually demonstrate better performance for image emotion recognition. In [65], Chen *et al.* proposed a visual sentiment concept classification method using deep convolutional neural networks. They showed that CNNs-based approaches significantly outperform SVM-based approaches for image sentiment annotation and retrieval. You *et al.* [66] proposed to progressively train convolutional networks on weakly labeled images to expand training data for sentiment analysis. The experimental results showed that this strategy effectively improves recognition accuracy. Rao *et al.* [67] proposed the MldrNet for image emotion classification. They claimed that their network can jointly learn image semantics, image aesthetics, and low-level visual features.

In [5], Pang *et al.* proposed a method to fuse multi-modal features, *i.e.,* visual and textual features, for image sentiment analysis, using the deep Boltzmann machine [68]. They demonstrated that the fused features are complimentary to single-modal features to improve performance for affective analysis and retrieval. However, using only fused features could not increase recognition accuracy compared to using single-modal features. In the of work of Yang *et al.* [69], a multi-task framework was developed for jointly emotion classification and emotion distribution regression. This framework can help to address the problem of annotating images with hard emotion labels. In [70], Zhu *et al.* proposed a unified CNN-RNN model, in which multi-level features are extracted from a CNN and are then integrated by a bidirectional recurrent neural network. Taking multi-level features into account, the CNN-RNN model effectively improves recognition accuracy. Pando *et al.* [71] proposed a curriculum guided strategy for learning discriminative emotion features. They demonstrated that the models that are trained on a large scale stock image dataset exhibit impressive generalization performance for image sentiment classification across datasets.

More recently, researchers started to research image emotion recognition from local regions instead of from whole images. In [7], Yang *et al.* proposed an approach to detect affective regions. The CNN features of the detected regions together with the whole image are aggregated to represent the whole image feature. In this method, thousands of

candidate regions are required to be processed for one input image, which is computationally intensive and time consuming. In [8], Rao *et al.* proposed a multi-level region-based convolutional neural network based on the feature pyramid network (FPN) for emotion classification in images. This network can detect discriminative emotion regions and has shown to be effective compared to conventional deep networks that have been developed for image classification. While this network is trained with only frame-level labels, it needs to be pretrained on a dataset annotated with region information.

Chapter 3

Recognition of Emotions in User-generated Videos with Kernelized Features

3.1 Introduction

Numerous video clips generated by users are uploaded to the Internet on a daily basis, thus highlighting the importance of developing intelligent algorithms for automatically understanding the content of these videos. In addition to describing different semantics, such as marriage proposals, award ceremonies and birthday parties, these videos may convey a number of emotions, including 'surprise' and 'joy'. Recognition of emotions in videos plays an important role in video understanding, and has many potential applications, such as video annotation, retrieval and recommendation. Over the years, recognition of emotions in user-generated videos has attracted increasing interests in the multimedia and computer vision community.

While significant progress has been made in recognizing semantics in videos, *e.g.*, action recognition [72–74] and event detection [75,76], little progress has been made regarding emotion recognition. Compared with recognition of semantics in videos, recognition of emotions is a complicated task. This is primarily due to the following two challenges: (1) Emotions have high intra-class variations. Video clips that convey a certain emotion might be captured in very different scenes. For example, video clips might capture the emotion of joy in a house in which a girl is playing a piano, in a stadium where people are watching a football game or on a lake where people are visiting a scenic spot. (2) Unlike video clips segmented from movies, user-generated videos are mostly taken by nonprofessional users with low-cost consumer cameras, such as mobile cameras and webcams. These video clips usually contain high levels of noise, such as image blur introduced by camera moving and low-resolution of economical cameras, making it difficult to recognize emotions in videos.

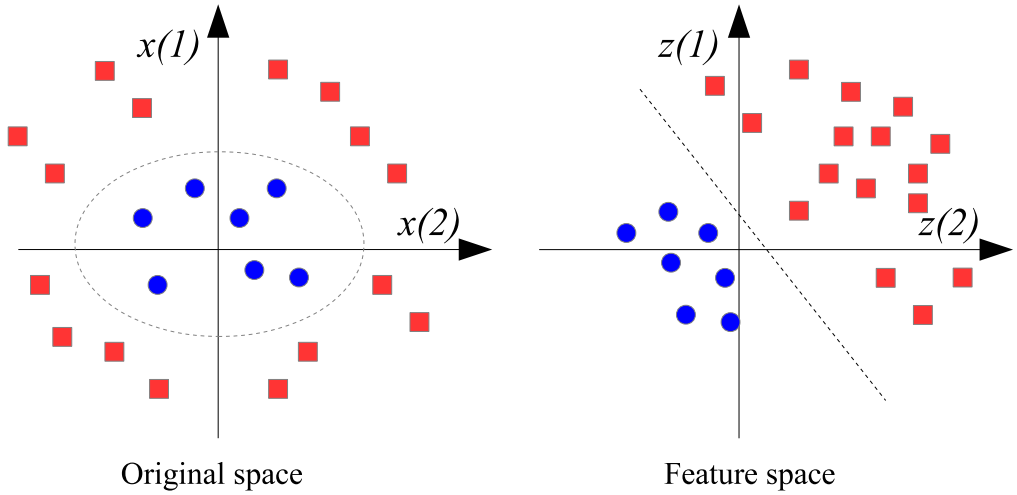


Figure 3.1 : An illustration of space transformation using a kernel function. The kernel function might not be explicitly written out.

Most existing approaches to the recognition of emotions in user-generated videos are based on advances in image representations. For example, Jiang *et al.* [48] proposed the use of attribute features, *e.g.*, ObjectBank [49] and SentiBank [64], for the recognition of emotions in user-generated videos. In attribute feature representation methods, an image is represented as a response map of numerous of pre-trained detectors. Compared with low-level image features, attribute features are interpretable and can model the semantic information of images. The experimental results showed that attribute features are effective for emotion prediction in user-generated videos. In contrast to hand-crafted features, deep features have shown success over the last few years due to their superior performance. Deep features extracted from the activations of a convolutional neural network (CNN) pre-trained on a large dataset have achieved state-of-the-art results on many visual recognition tasks, such as image classification [26,77], object detection [78,79] and image retrieval [80]. In the work of Xu *et al.* [54], deep features extracted from four pre-trained CNN models were evaluated for the recognition of emotions in user-generated videos. Compared with hand-crafted features, deep features achieved high recognition accuracy.

Leveraging the latest advances of CNNs for feature representations could improve recognition accuracy. This helps solve the first challenge to some extent. However,

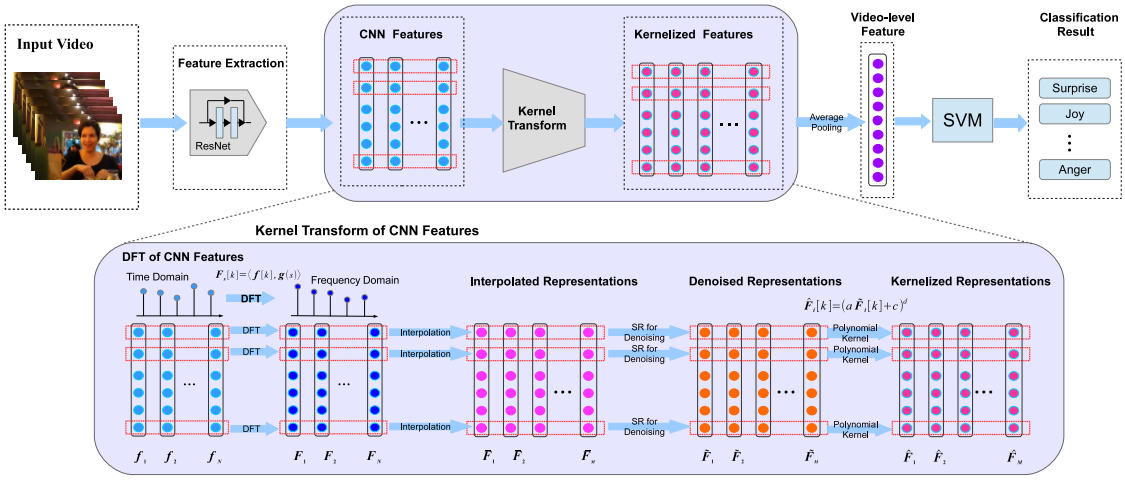


Figure 3.2 : An overview of the proposed approach for recognition of emotions in user-generated videos.

even with features extracted from state-of-the-art deep models, *e.g.* AlexNet [25], VGGNet [26], and GoogleNet [53], recognition accuracy is still very low, which is evidenced by the experimental results reported in [54], wherein utilizing deep features on two benchmark datasets, *i.e.*, VideoEmotion-8 [48] and Ekman-6 [54], could only obtain 43.8% and 50.9% recognition accuracy, respectively. This result is mainly due to the broad affective gap between image features and high-level emotions.

To the best of our knowledge, existing work on emotion recognition in videos has not considered reducing the impact of noise to deal with the second challenge. Denoising methods usually require optimization procedures. Because even a very short video clip may contain hundreds of frames, applying denoising methods would be extremely computationally extensive. Without considering suppressing the impact of noise in videos, the performance of existing approaches is restricted.

In this chapter, we aim to develop an effective video representation method. We propose to transform frame-level spatial features to another feature space with a kernel function to recognize emotions in videos. The intuition of the proposed approach is that the high-level discriminative emotion information of videos could be captured in another feature space. An illustrative example is shown in Figure 3.1. We construct a polynomial

kernel, which is inspired by the discrete Fourier transform (DFT). Moreover, we adopt the interpolation method to frequency domain signals and apply the sparse representation method for denoising.

The proposed approach involves four steps. An overview of the proposed approach is shown in Figure 3.2. First, frame-level features are extracted from the activations of a pre-trained CNN and then placed sequentially. Second, we apply the kernel method to frame-level spatial features for space transformation. At this step, the spare representation method is applied to intermediate representations for denoising. Third, average pooling is applied to kernelized representations to generate video-level representations. Lastly, an SVM is trained to differentiate different classes of video emotions.

This chapter provides the following contributions:

- We construct a polynomial kernel by reformulating the equation of the DFT as a linear kernel function. Leveraging the polynomial kernel, we propose to recognize emotions in user-generated videos with kernelized features. Compared with spatial image features, kernelized features show superior discriminative capability.
- We are the first to apply the sparse representation method to reduce the impact of noise contained in video frames, thus helping contribute to performance improvement.
- The proposed approach is evaluated on two challenging benchmark datasets, *i.e.*, VideoEmotion-8 and Ekman-6, wherein it exceeds the previous state-of-the-art result.

3.2 The Proposed Approach

The proposed approach to recognizing emotions in user-generated videos consists of four steps, which are introduced in detail as follows.

3.2.1 Frame-level Feature Extraction

Recent advances in deep learning have shown that compared with hand-crafted features, CNN features, which are extracted from the activations of a pre-trained CNN, are

much discriminative. In this work, we use CNN features for frame-level representations. In recent years, many CNN architectures, such as AlexNet, VGGNet, GoogleNet, and residual network (ResNet) [81], have been developed. According to our preliminary experimental results, better performance can be achieved using ResNet than the other three networks. Therefore, we utilize ResNet for frame-level feature extraction. ResNet has 152 layers and is pre-trained on ImageNet [24], which contains approximately 1.2 million training images labeled with 1,000 categories. This architecture contains many bottleneck building blocks. Each block comprises a stack of 1×1, 3×3, and 1×1 convolution layers, followed by a global average pooling layer, denoted as pool_5 following the notation used in [81], and a final 1000-way softmax. The pool_5 layer is the only fully connected layer before the softmax layer in this architecture. Considering features extracted from fully connected layers contain high-level information of input images [82], activations from the pool_5 layer are extracted for frame-level representations in this work.

After obtaining all frame-level features of a video clip, we place them sequentially in a table. Let $\mathbf{f}$ denote the sequentially placed frame-level features of a video clip with N frames, then $\mathbf{f}$ can be represented as follows:

$$\mathbf{f} = (\mathbf{f}_1, ..., \mathbf{f}_N) = \begin{bmatrix} \mathbf{f}_1[1] & \dots & \mathbf{f}_i[1] & \dots & \mathbf{f}_N[1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{f}_1[k] & \dots & \mathbf{f}_i[k] & \dots & \mathbf{f}_N[k] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{f}_1[D] & \dots & \mathbf{f}_i[D] & \dots & \mathbf{f}_N[D] \end{bmatrix}, \tag{3.1}$$

where $\mathbf{f}_i = (\mathbf{f}_i[1], ..., \mathbf{f}_i[D])^T$ represents the feature of the i-th frame with dimension D. In this work, D equals the dimension of the pool_5 layer of ResNet. The value of N is different for different video clips as their lengths vary.

3.2.2 Apply Kernel Method to CNN Features

Kernel methods have been widely used in machine learning, such as support vector machine (SVM) [83], and the construction of graphical models [84]. In most applications, kernel methods are used for transforming a feature from the original space to another feature space. The transformed space could be of infinite dimension. Kernel functions (or

kernels for simplification) $K(\mathbf{x}, \mathbf{x}')$ can be considered as a similarity measure between two vectors. Formally, a kernel $K(\cdot, \cdot)$ is defined as an inner product $< \cdot, \cdot >_{\mathcal{H}}$ of two vectors in a Hilbert space $\mathcal{H}$ [85]. Given two vectors $\mathbf{x}$ and $\mathbf{x}'$, the kernel is calculated as follows:

$$K(\mathbf{x}, \mathbf{x}') =< \phi(\mathbf{x}), \phi(\mathbf{x}') >_{\mathcal{H}}, \tag{3.2}$$

where $\phi(\mathbf{x})$ is a mapping function. In most cases, $\phi(\mathbf{x})$ can not be written out explicitly.

In Equation 3.1, each column vector represents a feature for a frame. In our work, we focus on developing an effective video representation method. To model the information among video frames, we apply the kernel method to row vectors of Equation (1) for space transformation. In most machine learning problems, algorithms can be expressed in terms of inner products between two vectors. In these cases, kernel functions can be applied directly. However, only one row vector is given in our problem. Another vector needs to be constructed to apply the kernel method.

Our method is inspired by the discrete Fourier transform (DFT), which is widely used in signal processing. The aim of the DFT is to transform a signal from the time domain to the frequency domain. Considering that the DFT can be used for space transformation, transformed signals might be obtained using a kernel function. We now take a look at the equation for the DFT, which is given below:

$$\mathbf{F}_s[k] = \sum_{n=1}^{N} \mathbf{f}_n[k] e^{-2i\pi(n-1)(s-1)/N}, \ s = 1, ..., N. \tag{3.3}$$

We observe that the above equation can be expressed as an inner product of two vectors as follows:

$$\mathbf{F}_s[k] =< \mathbf{f}[k], \mathbf{t} >, \ s = 1, ..., N \tag{3.4}$$

where $\mathbf{f}[k] = (\mathbf{f}_1[k], \mathbf{f}_2[k], ..., \mathbf{f}_N[k])^T$ and $\mathbf{t} = (1, e^{-2i\pi(s-1)/N}, ..., e^{-2i\pi(N-1)(s-1)/N})^T$. From the perspective of kernel methods, $\mathbf{F}_s[k]$ is obtained with a linear kernel over $\mathbf{f}[k]$ and $\mathbf{t}$.

According to Equation (3.3), the number of points obtained in the frequency domain is the same as that in the time domain. The computational complexity of the DFT is

$O(N^2)$. The fast Fourier transform (FFT) can compute the same results more quickly than by using Equation (3.3). With the FFT, the computational complexity is reduced to $O(NlogN)$. The computed value $\mathbf{F}_s[k] \in \mathbf{F}[k]$ is a complex number, and its absolute value represents the amplitude of the s-th frequency. In our work, the absolute value of $\mathbf{F}_s[k]$ is used instead of its original complex value. After applying the FFT to all row vectors of Equation (3.1), we obtain the following frequency feature table:

$$\mathbf{F} = (\mathbf{F}_1, ..., \mathbf{F}_N) = \begin{bmatrix} \mathbf{F}_1[1] & ... & \mathbf{F}_i[1] & ... & \mathbf{F}_N[1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{F}_1[k] & ... & \mathbf{F}_i[k] & ... & \mathbf{F}_N[k] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{F}_1[D] & ... & \mathbf{F}_i[D] & ... & \mathbf{F}_N[D] \end{bmatrix}, \tag{3.5}$$

where $\mathbf{F}_i = (\mathbf{F}_i[1], ..., \mathbf{F}_i[D])^T$, termed as a DFT feature in this thesis.

We have seen from Equation (3.3) that $\mathbf{F}_s[k]$ is computed with a linear kernel. We replace the linear product with a kernel function using the kernel trick [86], *i.e.*, wherever inner products are used, they can be replaced by kernel functions. There are many popular kernel functions, such as the polynomial kernel, Gaussian kernel and Laplace kernel. In our work, we choose to use the polynomial kernel because the computed frequency domain features can be utilized directly, whereas using other kernels could lead to higher computational complexity than using polynomial kernel. The equation for space transformation using the polynomial kernel is given below:

$$\begin{aligned} \breve{\mathbf{F}}_s[k] &= K(\mathbf{f}[k], \mathbf{t}) \\ &= (a < \mathbf{f}[k], \mathbf{t} > +c)^d \\ &= (a\mathbf{F}_s[k] + c)^d, \end{aligned} \tag{3.6}$$

where $c \geq 0$, and $d > 0$.

We have constructed a polynomial kernel to transform CNN features based on the DFT. The polynomial kernel can be applied to vectors with different lengths. After applying the polynomial kernel to CNN features in Equation (3.1), we obtain the following feature table:

$$\breve{\mathbf{F}} = (\breve{\mathbf{F}}_1, ..., \breve{\mathbf{F}}_N) = \begin{bmatrix} \breve{\mathbf{F}}_1[1] & ... & \breve{\mathbf{F}}_i[1] & ... & \breve{\mathbf{F}}_N[1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \breve{\mathbf{F}}_1[k] & ... & \breve{\mathbf{F}}_i[k] & ... & \breve{\mathbf{F}}_N[k] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \breve{\mathbf{F}}_1[D] & ... & \breve{\mathbf{F}}_i[D] & ... & \breve{\mathbf{F}}_N[D] \end{bmatrix}, \tag{3.7}$$

where $\breve{\mathbf{F}}_i = (\breve{\mathbf{F}}_i[1], ..., \breve{\mathbf{F}}_i[D])^T$, termed as a kernelized feature (obtained with a kernel function).

3.2.3 Sparse Representation for Denoising

As discussed in the introduction section, user-generated videos may contain high levels of noise. After applying the DFT to CNN features, we assume that the noise is transferred to the frequency domain, which makes the frequency distribution of the signals obtained in the frequency domain rough. In this work, we apply the denoising method to the frequency domain signals to improve the smoothness of the signals. Specifically, we adopt the sparse representation method, given its good denoising performance.

Let $\mathbf{x}$ be a row vector of $\mathbf{F}$ in Equation (3.7). Given a codebook with P entries, *i.e.*, $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_P]$, the sparse representation for $\mathbf{x}$ is obtained by solving the following optimization problem:

$$\begin{aligned} &\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_0 \\ &s.t.\ \|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_2 \leq \epsilon. \end{aligned} \tag{3.8}$$

In the above model, ℓ_0-norm represents the number of non-zero elements in a vector. The codebook $\mathbf{D}$ can be unsupervised learned by the K-means method. Once the sparse solution of the model (3.8) has been found, $\tilde{\mathbf{x}} = \mathbf{D}\hat{\boldsymbol{\alpha}}$ represents the denoised representation for $\mathbf{x}$.

Difficulty The model (3.8) requires that input vectors are of the same dimension. However, because different video clips have different lengths, *i.e.*, the dimension of $\mathbf{x}$ varies for different video clips, the model (3.8) can not be directly applied for solving sparse solutions for all video clips.

Solution To address this issue, we interpolate a fixed number (L) of points evenly from

$\mathbf{x}$ for all video clips. For example, given two video clips u and v with N and M frames, respectively, let $\mathbf{f}^u[k] = (\mathbf{f}_1^u[k], \mathbf{f}_2^u[k], ..., \mathbf{f}_N^u[k])$ and $\mathbf{f}^v[k] = (\mathbf{f}_1^v[k], \mathbf{f}_2^v[k], ..., \mathbf{f}_M^v[k])$ denote vectors of the k-th dimension of CNN features for u and v. We use Δt to represent the time interval between video frames, where Δt is a fixed value for all video clips. After transforming $\mathbf{f}^u[k]$ and $\mathbf{f}^v[k]$ to the frequency domain, we obtain their frequency distributions, *i.e.*, $\mathbf{F}^u[k] = (\mathbf{F}_1^u[k], \mathbf{F}_2^u[k], ..., \mathbf{F}_N^u[k])$ and $\mathbf{F}^v[k] = (\mathbf{F}_1^v[k], \mathbf{F}_2^v[k], ..., \mathbf{F}_M^v[k])$. Then, the model (3.8) can be applied for denoising.

From the perspective of Fourier analysis, points obtained in the frequency domain are evaluated over the range from 0 to S, where S equals the reciprocal of the sampling rate, *i.e.*, $S = 1/\Delta t$. As mentioned earlier, the number of points obtained in the frequency domain is the same as that in the time domain. Therefore, if $N > M$, the points of $\mathbf{F}^u[k]$ are spaced closer than those of $\mathbf{F}^v[k]$. After interpolation, we obtain $\mathbf{F}^u[k] = (\mathbf{F}_1^u[k], \mathbf{F}_2^u[k], ..., \mathbf{F}_L^u[k])$ and $\mathbf{F}^v[k] = (\mathbf{F}_1^v[k], \mathbf{F}_2^v[k], ..., \mathbf{F}_L^v[k])$, each with L points. Then, the two interpolated signals are evenly spaced by S/L and are evaluated over the same range from 0 to S. An example of the interpolation is shown in Figure 3.3. From this figure, we can see that after interpolation, the two signals have the same number of points, which are spaced by the same interval.

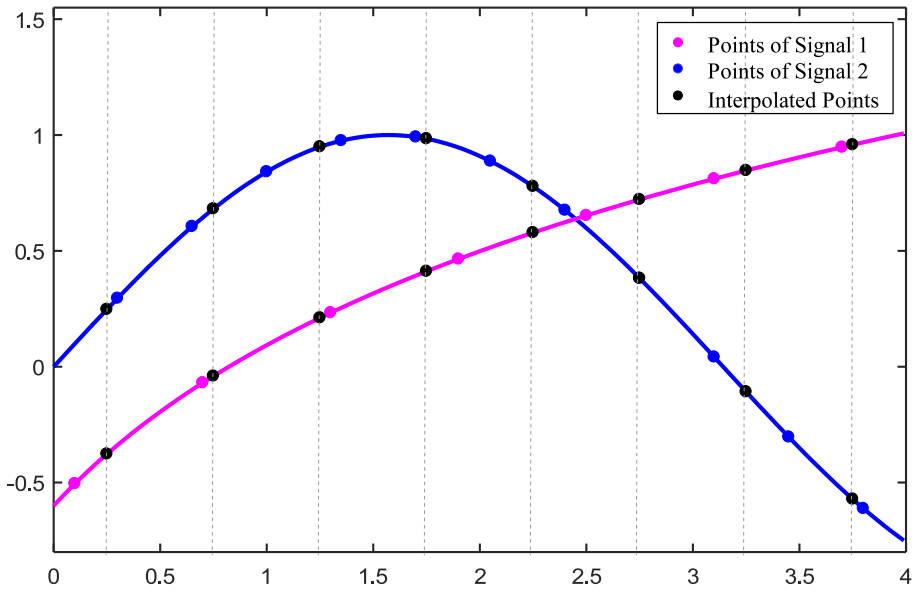


Figure 3.3 : An illustration of the effect of interpolation for two signals.

The optimization model (3.8) is non-convex and thus difficult to handle. The standard approach to solving the problem is to replace the ℓ_0-norm with the ℓ_1-norm, according to the theoretical results on the equivalence between ℓ_0-norm minimization and ℓ_1-norm minimization [87]. However, a computationally extensive procedure is required to solve the ℓ_1-norm optimization problem. To improve computational efficiency, we adopt the Locality-constrained Linear Coding (LLC) [88] method, which is an approximation method for computing sparse solutions. LLC utilizes locality constraints instead of sparsity constraints in the model (3.8) and can achieve much less quantization error than the model (3.8). In addition, LLC has an analytical solution; therefore, sparse solutions can be solved efficiently. After obtaining the sparse representation $\hat{\boldsymbol{\alpha}}$, the denoised signal is calculated as the multiplication of the dictionary $\boldsymbol{D}$ and $\hat{\boldsymbol{\alpha}}$. The denoised frequency domain features are given as follows:

$$\tilde{\mathbf{F}} = (\tilde{\mathbf{F}}_1, ..., \tilde{\mathbf{F}}_L) = \begin{bmatrix} \tilde{\mathbf{F}}_1[1] & ... & \tilde{\mathbf{F}}_i[1] & ... & \tilde{\mathbf{F}}_L[1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{\mathbf{F}}_1[k] & ... & \tilde{\mathbf{F}}_i[k] & ... & \tilde{\mathbf{F}}_L[k] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{\mathbf{F}}_1[D] & ... & \tilde{\mathbf{F}}_i[D] & ... & \tilde{\mathbf{F}}_L[D] \end{bmatrix}, \tag{3.9}$$

where $\tilde{\mathbf{F}}_i = (\tilde{\mathbf{F}}_i[1], ..., \tilde{\mathbf{F}}_i[D])^T$ represents a denoised DFT feature.

We use denoised values instead of their original values for space transformation with the polynomial kernel. The polynomial kernel function is given as follows:

$$\hat{\mathbf{F}}_s[k] = (a\tilde{\mathbf{F}}_s[k] + c)^d. \tag{3.10}$$

After applying Equation (3.10) to all denoised DFT features, we obtain the following kernelized feature table:

$$\hat{\mathbf{F}} = (\hat{\mathbf{F}}_1, ..., \hat{\mathbf{F}}_N) = \begin{bmatrix} \hat{\mathbf{F}}_1[1] & ... & \hat{\mathbf{F}}_i[1] & ... & \hat{\mathbf{F}}_L[1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \hat{\mathbf{F}}_1[k] & ... & \hat{\mathbf{F}}_i[k] & ... & \hat{\mathbf{F}}_L[k] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \hat{\mathbf{F}}_1[D] & ... & \hat{\mathbf{F}}_i[D] & ... & \hat{\mathbf{F}}_L[D] \end{bmatrix}, \tag{3.11}$$

where $\hat{\mathbf{F}}_i = (\hat{\mathbf{F}}_i[1], ..., \hat{\mathbf{F}}_i[D])^T$.

3.2.4 Video-level Representation and Classification

Given the kernelized features, average pooling is applied in this work for generating video-level representations. The equation for computing average pooling is as follows:

$$\mathbf{Fea}_{\text{video}} = \frac{1}{\mathrm{L}} \sum_{\mathrm{i}=1}^{\mathrm{L}} \hat{\mathbf{F}}_{\mathrm{i}}. \tag{3.12}$$

From Equation (3.4) and Equation (3.6), we can see that a kernelized feature is obtained taking into account information from all frames in a video clip. The intuition behind using average pooling is that in the space transformed by the polynomial kernel, each feature can contribute equally to video-level representation. All video-level representations are further ℓ_2 normalized. The dimension of the video-level feature $\mathbf{Fea}_{\text{video}}$ is the same as that of $\hat{\mathbf{F}}_i$. After obtaining video-level features, we train an SVM with a linear kernel to classify the video-level features into different emotion categories.

3.3 Experimental Results

3.3.1 Experimental Setup

To evaluate the performance of the proposed approach, experiments were conducted on two benchmark datasets, VideoEmotion-8 and Ekman-6.

VideoEmotion-8 Dataset [48]. This dataset contains 1,101 user-generated videos collected from popular video-sharing websites, *i.e.*, Youtube and Flicker. These videos are labeled with eight basic human emotion categories according to Plutchik's theory [13] on basic human emotions. There are at least 100 videos in each category. The average duration of the 1,101 videos is 107 seconds. Following [48], experiments were conducted ten times. In each time, we randomly selected $2/3$ of the data from each category for training, and the rest were used for testing. The average value of prediction accuracies from the ten experiments was calculated to evaluate the recognition performance. Because the similarity between adjacent video frames is very high, we evenly sampled one frame from every eight frames for computation efficiency.

Ekman-6 Dataset [54]. As with the VideoEmotion-8 dataset, this dataset was collected from social video-sharing websites (*i.e.*, YouTube and Flickr). In this dataset, there are 1,637 videos. These videos are labeled with six emotion categories based on Ekman's theory [12] on basic human emotion categories, with a minimum of 221 videos per category. The average duration of these video clips is 112 seconds. The train/test split is provided by the dataset. The training set contains 819 videos, and the testing set contains 818 videos. Classification accuracy is reported for performance evaluation. For computational efficiency, we evenly sampled one frame from every five frames.

Implementation Details In our experiments, frame-level features were extracted using the Caffe toolkit [89]. Specifically, all selected video frames were resized with the size of the shorter side equal to 256 while maintaining the original aspect ratio. For each selected frame, we cropped 10 regions of the size of 224×224 (four corners and one center, and their horizontal flip). The 10 crops were fed to ResNet, and activations of the $pool_5$ layer were extracted. The average of the 10 activations was calculated as the frame-level representation. All frame-level representations were ℓ_2 normalized. The FFT [90] was adopted for computing the DFT. At the denoising step, the cubic interpolation method [91] was adopted to generate a fixed length representation for all signals transformed by the DFT. After obtaining video-level representations, a multi-class SVM was trained using the LibLinear toolbox [92], which only supports linear kernels, to classify the video-level representations into different emotion categories.

3.3.2 Results on VideoEmotion-8

Results of the Proposed Approach

In our experiments, the values of a, c, and d for the polynomial kernel were empirically set to 4, $1/4$, and $5/4$, respectively. At the denoising step, the value of L was set to 700, and a dictionary with 1,024 entries was learned using the K-means method for the LLC encoding. The overall recognition accuracy of the proposed approach is shown in Table 3.1.

From this table, we can see that DFT features achieve a 1.7% performance gain as compared to CNN features, improving the recognition accuracy from 45.5% to 47.2%.

Table 3.1 : Overall recognition accuracy of the proposed approach on VideoEmotion-8.

Method	Overall accuracy (%)
CNN	45.5
DFT	47.2
Polynomial kernel	48.3
Polynomial kernel + Denoising	**49.7**

Utilizing kernelized features transformed by the polynomial kernel, we achieve 48.3% recognition accuracy, which is 1.1% higher than that using DFT features. After applying the sparse representation method for denosing, we achieve 49.7% recognition accuracy. Denoising yields a 1.4% performance improvement. Overall, kernelized representations with denoising contribute to a 4.2% performance gain compared with CNN features. Therefore, the proposed approach is effective for performance improvement.

We have seen that compared with CNN features, kernelized features improve the overall recognition accuracy. We then analyzed the recognition accuracy for each emotion category. The results are shown in Figure 3.5. We observe that the emotion of surprise has the highest recognition accuracy among the eight emotion categories. This might be because the 'surprise' category has the highest number of samples (236), whereas the 'joy' category, which has the second highest number of samples, contains only 180 samples. The emotions of anticipation and trust are difficult to recognize. This difficulty might be due to their very abstract concepts, and thus, they have no clear visual clues compared with the other categories.

In comparison with CNN features, DFT features improve recognition accuracy for all the emotion categories other than 'fear'. Except for 'joy' and 'surprise', the pure kernelized features (without applying the sparse representation method for denoising) achieve higher recognition accuracy than DFT features. Compared with the pure kernelized features, denoising can only improve recognition accuracy for half of the emotion categories, *i.e.*, 'anger', 'disgust', 'fear', and 'surprise'. For the other four categories, denoising degrades recognition accuracy. This result may be due to lost information after the in-

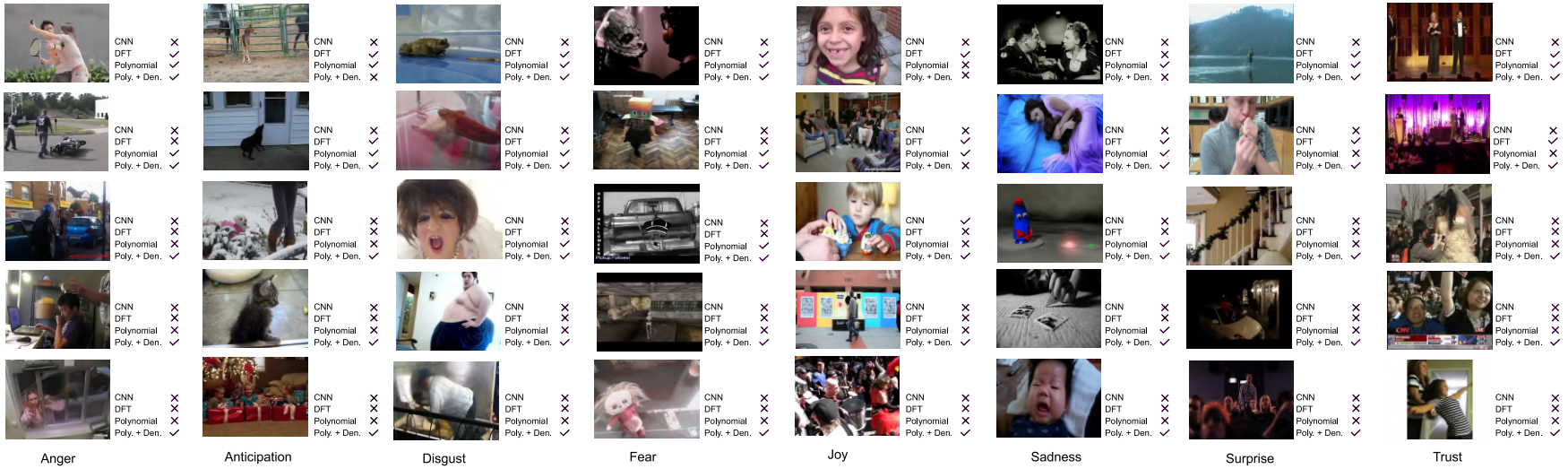


Figure 3.4 : Examples of recognition accuracy for each emotion category on VideoEmotion-8. A check mark (✓) represents a correct recognition result, while an X mark (×) represents an incorrect recognition result.

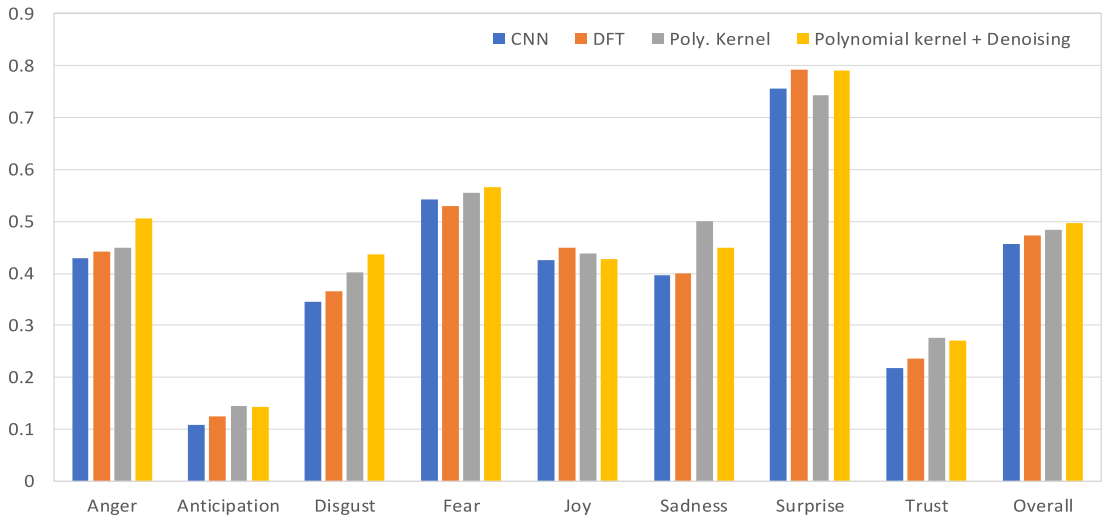


Figure 3.5 : Recognition accuracy for each emotion category on VideoEmotion-8 using the proposed approach.

terpolation step. Regardless, applying denoising helps improve the overall recognition accuracy. However, without interpolation, the sparse representation method cannot be applied. Overall, compared with CNN features, kernelized representations with denoising improve recognition accuracy for all emotion categories. Figure 3.4 shows examples of prediction results of the proposed approach for each category.

Analysis of Denoising: We have already seen that applying the sparse representation method for denoising helps improve the overall recognition accuracy. At the denoising step, L points are uniformly interpolated for each signal transformed by the DFT. We analyzed the impact of the number of interpolated points L on the overall recognition accuracy (see Table 3.2). For the LLC encoding, a dictionary with 1,024 entries was learned using the K-means method. The values of a, c, and d for the polynomial kernel were set to 4, $1/4$, and $5/4$, respectively.

From Table 3.2, we observe that the overall recognition accuracy improves as L increases. When L equals 700, the highest recognition accuracy is obtained. The recognition accuracy cannot be further improved when L is larger than 700. This might be because when L is set to 700, interpolated signals are good approximations to the original

Table 3.2 : Impact of the number of interpolated points on the overall recognition accuracy.

L	300	500	700	900
Overall accuracy(%)	48.4	49.0	**49.7**	49.2

signals.

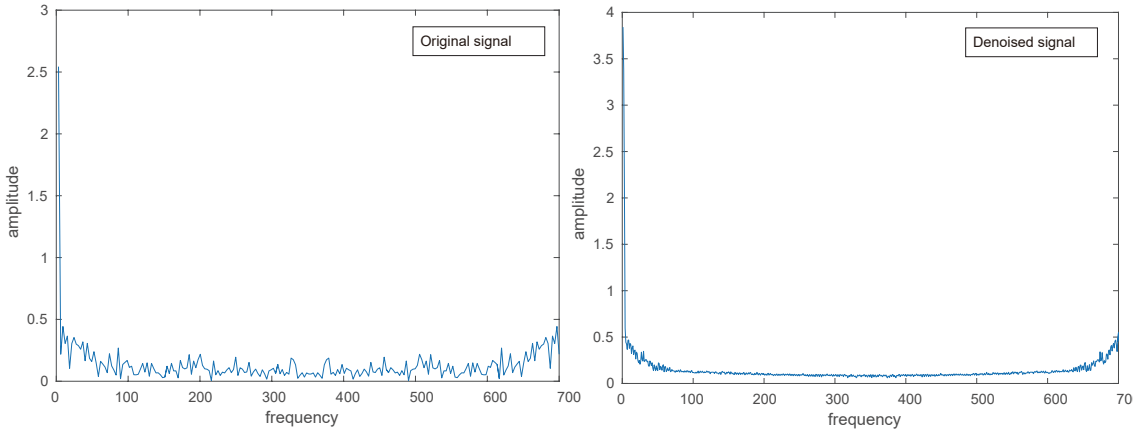


Figure 3.6 : Illustration of the effect of denoising using LLC for a signal in the frequency domain.

Figure 3.6 shows an example of the effect of denoising for a frequency domain signal in our experiments. We observe that the frequency distribution becomes much smoother after denoising. Compared with the recognition accuracy obtained with pure kernelized representations, denoising yields a 1.4% performance gain, thus improving the recognition accuracy from 48.3% to 49.7%. We conclude that applying the sparse representation method for denoising makes kernelized features more discriminative.

We further compared the performance of the proposed approach using features extracted from different CNN architectures. We used AlexNet, VGG-16, and VGG-19, which are pre-trained on ImageNet, and extracted the activations of the fc_7 layer as video-level features. The comparison results are shown in Table 3.3.

From the above table, we find that the polynomial kernel improves the classification

Table 3.3 : Performance of the proposed approach using features extracted from different CNN architectures on VideoEmotion-8.

Feature type	CNN	DFT	Polynomial kernel	Polynomial kernel + Denoising
AlexNet (fc_7)	43.0	43.3	45.1	45.6
VGG-16 (fc_7)	44.8	45.3	46.0	46.7
VGG-19 (fc_7)	44.0	44.7	45.0	47.2
ResNet-152 ($pool_5$)	**45.5**	**47.2**	**48.3**	**49.7**

accuracy for the four types of CNN features. The denoising step could further improve performance. This shows that our approach is stable for different CNN features. The ResNet features achieve the best performance among the four types of CNN architectures. One possible reason is that the dimension of the $pool_5$ layer in ResNet is 2048, which is half of the dimension of the fully connected layers in VGGNet and AlexNet. Because features with large dimensions may lead to classifier over-fitting, the ResNet features achieve better performance than the other three types of CNN features.

Previous research has shown that fine-tuning CNNs, which are pre-trained on ImageNet, on other visual recognition tasks, such as object recognition and scene classification, could improve recognition performance. To compare with our method, we fine-tuned the ResNet-152 on video frames of the training dataset. Video-level labels were used as frame labels when fine-tuning the network. We used the stochastic gradient descent (SGD) algorithm with a mini-batch size of 8. The learning rate started from 0.1 and was divided by 10 after every 100,000 iterations. The models were trained for up to 380,000 iterations. The values of weight decay and momentum were set to 0.0001 and 0.9, respectively. We used the $pool_5$ features extracted from the fine-tuned ResNet-152 to evaluate the performance of the proposed approach. Experimental results are shown in Table 3.4.

We observe that using features extracted from the ResNet fine-tuned on video frames could not improve classification accuracy. This observation is the same as the results reported in [51], wherein the authors reported that fine-tuning CNNs on video frames

Table 3.4 : Results of the proposed approach using features extracted from the ResNet fine-tuned on video frames on VideoEmotion-8.

Feature type	CNN	DFT	Polynomial kernel	Polynomial kernel + Denoising
ResNet (fine-tuned)	37.6	38.6	38.8	40.3

could not improve recognition accuracy on this dataset. Unlike emotion recognition in images, frames in a video clip are combined together to reflect the dominant emotion of a video clip. Many frames in a video clip may convey a neutral emotion or convey an emotion other than the dominant emotion. Therefore, video-level labels are extremely weak at the frame level, and thus the recognition accuracy cannot be improved.

In [93], the authors reported that using feature pooling methods, such as Fisher Vector (FV) [94] and Vector of Locally Aggregated Descriptors (VLAD) [95], could achieve better performance than using average pooling for event detection in videos. In this work, we further conducted experiments using the two pooling methods. We applied FV and VLAD to denoised kernel features. The number of Gaussian components for FV was set to 2, and the number of clusters for VLAD was set to 64. The appendix section presents the impact of the number of Gaussian components for FV and the number of clusters for VLAD on classification accuracy. The comparison results are shown in Table 3.5.

Table 3.5 : Performance of the proposed approach using different pooling methods on VideoEmotion-8.

Pooling method	Overall accuracy (%)
Average pooling	**49.7**
FV	46.0
VLAD	41.6

From Table 3.5, we observe that average pooling achieves the best performance among the three pooling methods. In previous work, such as [93] and [95], FV and VLAD

were claimed to be good pooling methods for an overcomplete feature set (*i.e.*, the number of video frames is much larger than the dimension of frame-level features) [96]. In our work, pooling methods are applied to a non-overcomplete feature set (see Equation (3.11), *i.e.*, the number of video frames is approximately 400, which is much less than the dimension of frame-level features (2048)). This might be a reason why average pooling performs well in our case, compared with FV and VLAD. Another reason might be related to the large dimension of features generated by FV and VLAD. Let D represent the dimension of frame-level features, and the dimension of video-level features generated by FV and VLAD are $2C_GD$ and C_KD, respectively, where C_G is the number of Gaussian components learned by the GMM model and C_K is the number of clusters learned by the K-means method. Features with large dimensions could lead to classifier over-fitting, therefore degrading the recognition accuracy using FV or VLAD.

Comparison with Previous Work

To demonstrate the advantage of the proposed approach, we compared the performance of the proposed approach with the recent work on VideoEmotion-8. The comparison results are shown in Table 3.6.

Table 3.6 : Comparison with previous work on VideoEmotion-8.

Method	Overall accuracy(%)
SentiBank [64]	35.5
EMDBM [5]	40.4
$\mathrm{CNN}_{\mathrm{VGG16}}$ [54]	44.7
ITE [54]	44.7
Our approach	**49.7**
Vis.+Aud.+Att. [48]	46.1
Context Fusion Net [51]	50.4
Vis.+Aud.+Att. +EMDBM [5]	51.1
Our approach + SentiBank	**52.5**

With a single-type feature extracted from ResNet, we obtain a recognition of 49.7% accuracy using the polynomial kernel for space transformation with the sparse representation method for denoising. When compared with SentiBank, the proposed approach exhibits 14.2% improvement. This result shows the advantage of kernelized features over attribute features. Our approach outperforms EMDBM and ITE by 9.3% and 5.0%, respectively. In EMDBM, an unsupervised method was proposed using the deep Boltzmann machine for learning video representations. This method depends on a large auxiliary image and video dataset for unsupervised training. In addition, a computationally extensive optimization procedure is required to train the model. In ITE, the authors used transfer learning to learn video representations from an additional image dataset. In contrast, we only used training data from the VideoEmotion-8 dataset in our work.

As mentioned earlier, emotions have high intra-class variations. In previous work, multiple features were utilized to achieve a good performance. When we combine the kernelized representations with SentiBank, we obtain an overall recognition accuracy of 52.5%. When compared with [48], in which low-level visual, audio and attribute features were used, [48], our approach achieves a 6.4% improvement. This demonstrates the great advantage of kernelized features over hand-crafted features. Compared with [51] and [5], our method improves the results by 1.9% and 1.4%, respectively. In [5], features were extracted from four pre-trainied CNN architectures, *i.e.*, AlexNet, GoogleNet, VGG16, and VGG19, and were further fused by a neural network for generating video-level representations, whereas a single-type feature extracted from ResNet was used in our work. To the best of our knowledge, this is the best performance on the VideoEmotion-8 dataset.

3.3.3 Results on Ekman-6

Results of the Proposed Approach

For experiments on this dataset, the values of a, c, and d for the polynomial kernel were empirically set to 4, $1/4$, and $5/4$, respectively. At the denoising step, 500 points were evenly interpolated from each frequency domain signal, and a dictionary with 4,096 entries was learned using the K-means method for the LLC encoding. The overall recognition accuracy of the proposed approach is shown in Table 3.7.

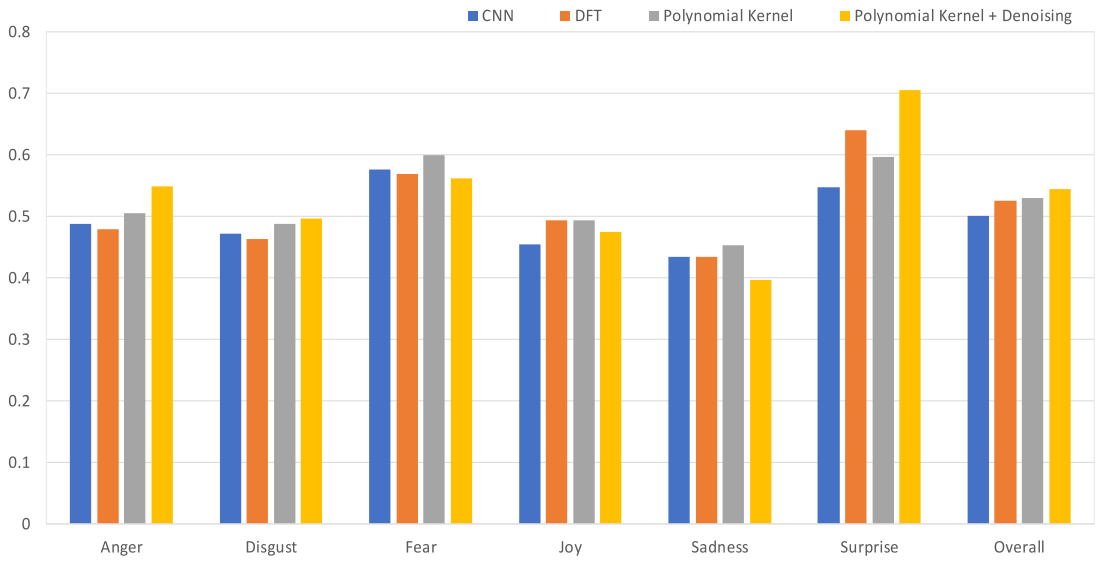


Figure 3.7 : Recognition accuracy for each emotion category on Ekman-6 using the proposed approach.

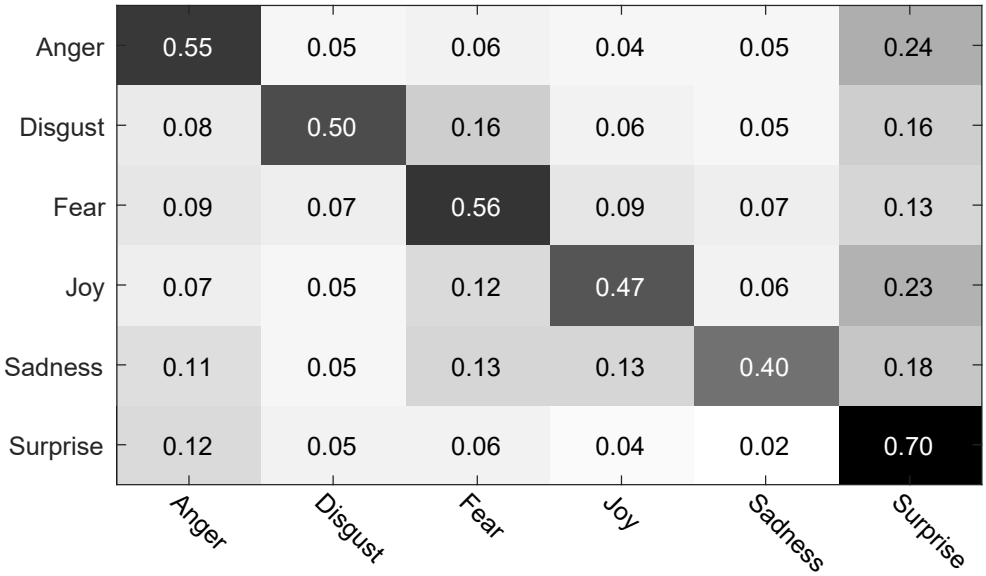


Figure 3.8 : Confusion matrix on Ekman-6 using kernelized features with denoising.

From Table 3.7, we find that compared with utilizing CNN features, utilizing DFT features improves the recognition accuracy from 50.0% to 52.1%. Compared with DFT features, using kernelized representations without denoising achieves a recognition accuracy

Figure 3.9 : Examples of recognition accuracy for each emotion category on Ekman-6. A check mark (✓) represents a correct recognition result, while an X mark (×) represents an incorrect recognition result.

Table 3.7 : Overall recognition accuracy of the proposed approach on Ekman-6.

Method	Overall accuracy (%)
CNN	50.0
DFT	52.1
Polynomian kernel	53.3
Polynomial kernel + Denoising	**54.4**

of 53.3%, yielding a 1.2% performance gain. After applying the sparse representation method for denoising, we achieve a recognition accuracy of 54.4%. Applying denoising contributes to a 1.1% performance gain. Compared with CNN features, using kernelized representations with denoising achieves a 4.4% performance improvement. This demonstrates the considerable advantage using kernelized features over CNN features.

Recognition accuracy for each emotion category of Ekman-6 is shown in Figure 3.7. We observe that while DFT features degrade the recognition accuracy for three emotion categories, *i.e.*, 'anger', 'disgust', and 'fear', DFT features help to improve the overall classification accuracy. Compared with DFT features, using kernelized features without denoising can achieve performance improvement for most emotion categories. Compared with CNN features, kernelized features without applying denoising improve the recognition accuracy for all emotion categories. Applying the sparse representation method for denoising results in performance improvement for only half of the emotion categories, *i.e.*, 'anger', 'disgust' and 'surprise'; however, it helps to improve the overall classification accuracy. Compared with CNN features, kernelized representations with denoising improve the overall recognition accuracy; however, it degrades recognition accuracy for the emotions of fear and sadness. Figure 3.9 shows examples of prediction results of the proposed approach for each category.

The confusion matrix of the recognition accuracy using kernelized features with denoising is shown in Figure 3.8. We observe that most categories are confused with the 'surprise' category. As with VideoEmotion-8, the proposed approach achieves the highest performance for the emotion of surprise. Although the 'surprise' category has the high-

est number of samples, the recognition accuracy for ‘surprise’ using CNN features is not the highest; however, using our approach significantly improves recognition accuracy for ‘surprise’. The proposed approach performs badly for the ‘sadness’ category. For this category, pure kernelized features achieve higher performance than CNN features and DFT features; however, applying denoising degrades the performance.

Table 3.8 : Performance of the proposed approach using features extracted from different CNN architectures on Ekman-6.

Feature type	CNN	DFT	Polynomial kernel	Polynomial kernel + Denoising
AlexNet (fc_7)	48.3	48.3	49.0	51.1
VGG-16 (fc_7)	49.6	49.9	50.4	50.9
VGG-19 (fc_7)	49.8	50.5	50.8	51.5
ResNet-152 ($pool_5$)	**50.0**	**52.1**	**53.3**	**54.4**

We compared the performance of the proposed approach using features extracted from different CNN architectures. We used the fc_7 layer features extracted from AlexNet, VGG-16 and VGG-19, which are pre-trained on ImageNet. The comparison results are shown in Table 3.8. From this table, we find that compared with CNN features, kernelized features improve the performance, and that the ResNet features achieve the best performance among the four types of CNN architectures.

Table 3.9 : Results of the proposed approach using features from the ResNet fine-tuned on video frames on Ekman-6.

Feature type	CNN	DFT	Polynomial kernel	Polynomial kernel + Denoising
ResNet (fine-tuned)	40.6	42.6	43.1	43.9

We have seen that using features extracted from ResNet achieves better performance than using features extracted from VGGNet and AlexNet. We then show the results using

features extracted from the ResNet fine-tuned on video frames of the training dataset. We used the SGD algorithm with a mini-batch size of 8. The learning rate started from 0.1 and was divided by 10 after every 100,000 iterations. The models were trained for up to 380,000 iterations. The values of weight decay and momentum were set to 0.0001 and 0.9, respectively. We used the $pool_5$ features extracted from the fine-tuned ResNet-152 to evaluate the performance of the proposed approach. The experimental results are shown in Table 3.9. We observe that using features extracted from the ResNet fine-tuned on video frames could not improve the recognition accuracy.

We further compared recognition accuracy using average pooling with FV and VLAD. The pooling methods were applied to denoised polynomial features. The number of Gaussian components for FV was set to 2, and the number of clusters for VLAD was set to 32. The appendix section presents the impact of Gaussian components for FV and the number clusters for VLAD on recognition accuracy. The comparison results are shown in Table 3.10.

Table 3.10 : Performance of the proposed approach using different pooling methods on Ekman-6.

Pooling Method	Overall accuracy (%)
Average pooling	**54.4**
FV	49.6
VLAD	53.0

From Table 3.10, we observe that compared with average pooling, FV and VLAD cannot improve classification accuracy. VLAD achieves 1.4% lower accuracy than average pooling. VLAD achieves higher performance on Ekman-6 than on VideoEmotion-8. This might be because there are more training samples for each category in Ekman-6 and increasing the number of training samples addresses the issue of classifier over-fitting to some extent.

Comparison with Previous Work

We compared our approach with two most recent work on this dataset. The comparison results are shown in Table 4.4.

Table 3.11 : Comparison with previous work on Ekman-6.

Method	Overall accuracy (%)
ITE [54]	50.4
Context Fusion Net [51]	51.8
Polynomial kernel + Denosing	**54.4**

Compared with image transfer encoding (ITE) and context fusion net, our approach achieves a 4.0- and 2.6-percentage improvement, respectively. In ITE, auxiliary images and text corpora were utilized by transfer learning for improving accuracy, whereas only training data of the Ekman-6 dataset were utilized in our work. In the context fusion net, three types of CNN features were utilized for emotion recognition, whereas a single-type feature extracted from the ResNet was used in our work. The comparison results demonstrate the advantage of the proposed approach. To the best of our knowledge, this is the best performance on Ekman-6.

3.4 Conclusion

In this chapter, we have proposed an approach for the recognition of emotions in user-generated videos. Our approach is based on kernelized features. To generate the kernelized features, we constructed a polynomial kernel function. Compared with spatial image features, kernelized features show superior performance. In addition, this is the first work to apply the sparse representation method for suppressing the impact of noise contained in video frames. On two challenging benchmark datasets, *i.e.*, VideoEmotion-8 and Ekman-6, the proposed approach has achieved state-of-the-art performance compared with that of previous methods.

The proposed approach can be generalized to other video classification tasks. In our

future work, the proposed method will be evaluated for other video classification tasks, such as action recognition and event detection.

3.A Appendix

The impact of the number of Gaussian components for FV on the overall recognition accuracy on VideoEmotion-8 is shown in Table 3.12.

Table 3.12 : Impact of the number of Gaussian components for FV on the overall recognition accuracy on VideoEmotion-8.

Number of components	2	4	8	16	32
Overall accuracy(%)	**46.0**	44.7	44.7	44.0	43.8

The impact of the number of clusters for VLAD on the overall recognition accuracy on VideoEmotion-8 is shown in Table 3.13.

Table 3.13 : Impact of the number of clusters for VLAD on the overall recognition accuracy on VideoEmotion-8.

Number of clusters	2	4	8	16	32	64	128
Overall accuracy(%)	29.9	30.5	35.8	36.2	39.3	**41.6**	41.5

The impact of the number of Gaussian components for FV on the overall recognition accuracy on Ekman-6 is shown in Table 3.14.

The impact of the number of clusters for FV on the overall recognition accuracy on Ekman-6 is shown in Table 3.15.

Table 3.14 : Impact of the number of Gaussian components for FV on the overall recognition accuracy on Ekman-6.

Number of components	2	4	8	16	32
Overall accuracy(%)	**51.7**	51.6	49.6	48.9	48.0

Table 3.15 : Impact of the number of clusters for VLAD on the overall recognition accuracy on Ekman-6.

Number of clusters	2	4	8	16	32	64	128
Overall accuracy(%)	48.1	51.3	52.1	52.0	**53.0**	52.9	52.4

Chapter 4

Frame-level Emotion Intensity Prediction for Improving Video Emotion Recognition Performance

4.1 Introduction

Compared to semantic recognition in videos such as human action recognition [72], video salient object detection [97], and event detection [93], little progress has been made regarding recognition of emotions in videos. Most existing approaches to semantic recognition in videos use deep learning technologies. Deep neural networks can automatically learn features from training data, which have shown to be much robust compared to conventional hand-crafted features. The development of deep technologies for semantic recognition in videos benefits a lot from publicly available large scale video benchmark datasets, such as the Sports-1M dataset [98] and the Youtube-8M dataset [99] which contain millions of video clips. However, for the video emotion recognition task, the two existing largest benchmark datasets , *i.e.,* VideoEmotion-8 [48] and Ekman-6 [54], only contain 1,101 and 1,637 video clips, respectively. This has restricted the development of deep learning technologies for video emotion recognition. Many deep neural networks developed for semantic recognition tasks can not be directly applied for video emotion recognition due to limited annotated data.

Currently, most methods for video emotion recognition involve three stages. First, frame-level features, *e.g.,* low-level features and deep features, are extracted. Then, the extracted frame-level features are aggregated with a feature pooling method to generate video-level features. Lastly, a classifier, *e.g.,* support vector machine (SVM), is trained on video-level features to classify testing videos into different emotion categories. The work of [54] has shown that using deep features, which are represented as the activations of a hidden layer from a convolutional neural network (CNN) pretrained on a large scale of image dataset, improves recognition performance over hand-crafted features. In the

work of [100], the authors showed that applying the average pooling method to deep features achieves better performance than other advanced feature pooling methods such as Fisher vector (FV) encoding and vector of locality aggregated vectors (VLAD) [95]. This might be because the number of feature dimensions generated by FV and VLAD is fairly large compared with that generated by average pooling; and this could lead to classifier overfitting when limited training data are available.

Unlike recognition of emotions in single images, a video clip contains a number of frames which are grouped together to convey a dominant emotion. In the average pooling method, each frame-level feature is equally taken into account to compute final video-level feature. However, some frames in the video clip may convey a neutral emotion other than the dominant emotion. These frames should be eliminated for computing video-level features. Even for frames that convey an emotions, the intensity of these frames conveying an emotion is different. They should not be equally taken into account to compute video-level features. Therefore, applying average pooling for computing video-level features could lead to the obtained features less robust.

To address this issue, we propose to use weighted sum pooling for computing video-level features in this thesis. Unlike the average pooling method, each frame-level feature is associated with an emotion intensity value in the proposed method. The weighted summation of frame-level features is computed as video-level representations (see Figure 4.1). The value of emotion intensity is predicted by a deep neural network. Our method for predicting emotion intensity is inspired by the class activation mapping (CAM) technique [10]. With the CAM technique, class-specific saliency regions can be derived from a deep convolutional neural network which is trained on image-level labels. We hypothesize that emotion intensity maps can be directly learned by deep networks. If the predicted emotion intensity map covers a large part of the image, this image has a high probability conveying an emotion. On the contrary, the image has a low probability conveying an emotion if the predicted intensity map covers a small part of the image.

Our network is first trained on a large scale of image dataset which was recently introduced for image emotion recognition. The CAM technique is used to generate saliency maps to guide the network for emotion intensity map learning. Then, we pass video

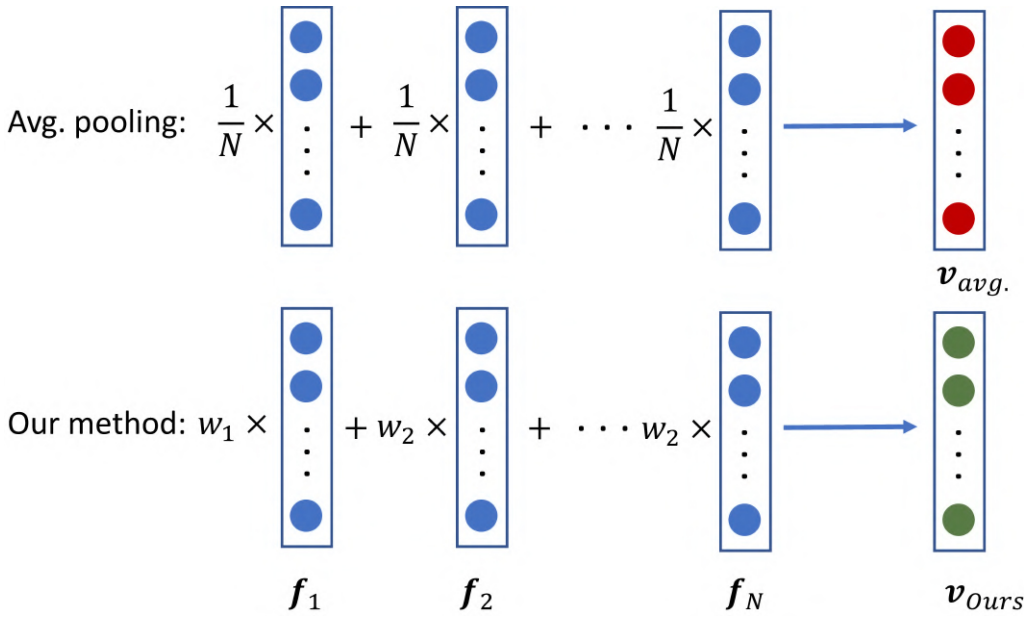


Figure 4.1 : Unlike average pooling, each frame-level feature is associated with an emotion intensity value in our method, the weighted summation is calculated as video features.

frames to the pretrained network and calculate the average values of the predicted intensity maps as frame-level intensity values for input frames. We empirically show that our approach achieve state-of-the-art performance on two emotion recognition benchmark datasets compared to previous methods.

4.2 Methodology

We aim to employ frame-level emotion intensity to improve video emotion recognition performance. To this end, an end-to-end network for simultaneous image emotion intensity prediction and classification is proposed. An overview of the proposed method for video representation is shown in Figure 4.2. In this section, we first present the proposed network using the CAM technique for weakly supervised emotion intensity prediction. Then, we introduce utilizing frame-level emotion intensity map for video feature generation and classification.

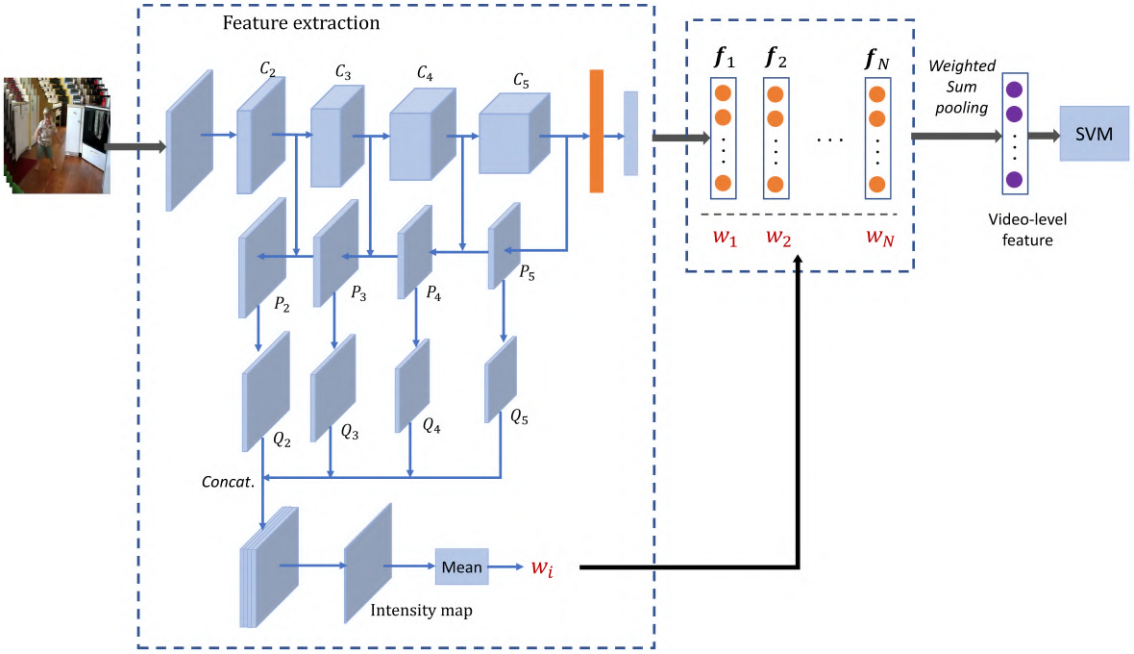


Figure 4.2 : An overview of the proposed approach for video representation and recognition. Each selected video frame is passed to a pretrained deep neural network. The activations before the last fully connected layer are extracted as a frame-level feature, and the average value of the predicted intensity map is calculated as the weight for the frame-level feature. The weighted sum pooling is applied to generate video-level features. Finally, an SVM is trained for prediction.

4.2.1 The network architecture

Our network for weakly supervised emotion intensity map learning is built based on FPN. With FPN, the network can learn multi-scale semantically strong features. The FPN architecture consists of a bottom-up pathway, a top-down pathway and and lateral connections. The bottom-up pathway is the feed-forward computation by the backbone network. It produces a feature hierarchy, which consists of feature maps at different scales with a scaling factor of 2. Let the activations of the last bottleneck layer of the four convolutional blocks be denoted as C_2, C_3,C_4, and C_5, respectively (see Figure 5.3). The feature pyramid can be represented as $\{C_2, C_3, C_4, C_5\}$. Following the work of [101], the feature maps generated by the first convolutional layer are not included to the feature

pyramid due to the large memory cost.

In the top-down pathway, higher resolution features is upsampled spatially coarser. Each lateral connection merges feature maps of the same spatial size from the bottom-up pathway and the top-down pathway. Specifically, we apply the upsampling operation with a factor of 2 (the bilinear interpolation is used in this work) to coarse-resolution feature maps. The upsampled feature maps is then merged with the corresponding bottom-up maps by element-wise addition. 1×1 convolutions with the output channel number equal to 256 are applied to bottom-up feature maps to reduce channel dimensions before addition.

After the top-down pathway, we obtain the set of feature maps $\{P_2, P_3, P_4, P_5\}$, each with 256 channels, corresponding to $\{C_2, C_3, C_4, C_5\}$. Each of the feature maps in $\{P_2, P_3, P_4, P_5\}$ is then connected to a stack of two 3×3 convolutional layers with the output channel number equal to 128. Then, we get the set of feature maps $\{Q_2, Q_3, Q_4, Q_5\}$, each with 128 channels, corresponding to $\{P_2, P_3, P_4, P_5\}$. The feature maps in $\{Q_2, Q_3, Q_4, Q_5\}$ are concatenated to generate 512 channels of feature maps. Because the size of the these feature maps is different, the bilinear interpolation method is employed before concatenation. This is further followed by two consecutive 3×3 convolutional layers, which end up with a sigmoid layer to produce emotion intensity maps. We do not use non-linearity in the top-down pathway. The size of outputs is $1/4$ of original input size. The average value of the predicted intensity map is calculated as the emotion intensity for the input image.

4.2.2 CAM guided Pseudo intensity map generation

Because we only have access to image-level annotations, we train the proposed network for emotion intensity prediction through weakly supervised learning. As introduced in Section 4.1, saliency regions can be derived from CNNs. In this work, we use the CAM method to generate pseudo intensity maps to guide the proposed network for emotion intensity learning. Specifically, The ResNet [27] architecture is used as the backbone network. The global average pooling layer outputs the spatial average of the feature maps produced by the last convolutional layer. Following the work of Zhou *et al.* [10], the

bias term in the last layer is ignored as it has trivial impact on classification performance. The CAM technique identifies the saliency regions by projecting back the weights of the output layers on to the convolutional feature maps. The class activation map is represented as the weighted linear sum of the presence of these visual patterns at different spatial locations (see Equation 5.1).

The values of the obtained pseudo saliency maps are rescaled to 0-255 as the emotion intensity representation. We further upsampling the obtained intensity maps to the desired size using the bilinear interpolation method to guide our network for supervised intensity map learning. The value of an emotion intensity map represents to what degree this area represents an emotion. Intensity maps provide discriminative local information that can be used to improve recognition performance.

4.2.3 The loss functions

In this work, we investigate three loss functions, *i.e.,* root mean square error in log space, gradient loss, and surface normal loss, on intensity map prediction performance. The utilization of the three loss functions is inspired by the work on image depth prediction. The details of the three loss functions are introduced as follows.

1) Root mean square error in log space (RMSEL) [102]. The RMSEL loss function empirically achieves better prediction performance than root mean square error (RMSE), $L1$-norm loss, and $L2$-norm loss. The RMSEL loss function is defined as follows:

$$L_{RMSEL} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sqrt{(log(p_{ij}) - log(g_{ij}))^2}, \tag{4.1}$$

where p_{ij} and g_{ij} denote the values of the predicted intensity map and the pseudo intensity map generated by the CAM technique, respectively, at the spatial location (i, j) and N represents the height and width of the predicted intensity map.

2) Gradient loss The gradient of an intensity map $\boldsymbol{f}$ is given as follows:

$$\nabla \boldsymbol{f} = [\frac{\partial \boldsymbol{f}}{\partial x}, \frac{\partial \boldsymbol{f}}{\partial y}], \tag{4.2}$$

where $\frac{\partial \boldsymbol{f}}{\partial x}$ and $\frac{\partial \boldsymbol{f}}{\partial y}$ denote the partial derivative of the intensity map with respective to the x direction and the y direction, respectively. In this work, the gradient of intensity maps

is obtained by applying two Sobel filters [103]. The gradient loss is the L_1 norm of the difference between the norm of the predicted intensity maps, which is given as follows:

$$L_{gradient} = \frac{1}{N}\sum_{i=1}^{N}\sum_{i=1}^{N}\left\|\nabla d_{ij} - \nabla p_{ij}\right\|_1, \tag{4.3}$$

where $\|\cdot\|$ denotes L_1-norm, and ∇d_{ij} and ∇p_{ij} denote the gradients of the predicted intensity map and of the intensity map generated by the CAM method computed at the spatial location (i, j), respectively.

3) Surface normal loss The surface normal loss [104] measures the accuracy of the normal to the surface of an estimated intensity map with respect to its ground truth. The surface normal loss is represented as follows:

$$L_{normal} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{i=1}^{N}(1 - \frac{\langle n_{ij}^d, n_{ij}^g\rangle}{\sqrt{\langle n_{ij}^d, n_{ij}^d\rangle}\sqrt{\langle n_{ij}^g, n_{ij}^g\rangle}}), \tag{4.4}$$

where and $\langle\cdot,\cdot\rangle$ denotes inner product of two vectors, which measure the angle between two surface normals, $n_{ij}^d = [-\nabla_x(d_{ij}), -\nabla_y(d_{ij}), 1]$ and $n_{ij}^g = [-\nabla_x(g_{ij}), -\nabla_y(g_{ij}), 1]$ denotes the surface normals of the predicted intensity map and the intensity map generated by the CAM method, respectively, at the spatial location (i, j).

The gradient loss and the surface normal loss help to refine predicted intensity maps. The overall emotion intensity loss function is defined as follows:

$$\begin{aligned} L_{intensity} &= \lambda_1 L_{RMSEL} + \lambda_2 L_{normal} \\ &\quad + \lambda_3 L_{gradient}, \end{aligned} \tag{4.5}$$

where λ_1, λ_2, and λ_3 are hyperparameters.

The overall network is trained with the multi-task loss, including the classification loss and the intensity map prediction loss, which as follows:

$$\begin{aligned} L &= \lambda_0 L_{cls} + L_{intensity} \\ &= \lambda_0 L_{cls} + \lambda_1 L_{RMSEL} + \lambda_2 L_{normal} \\ &\quad + \lambda_3 L_{gradient}, \end{aligned} \tag{4.6}$$

where λ_0 represents the hyperparameter for the classification loss. In this work, the cross entropy loss is used as the classification loss. The intensity prediction network can be seen as a regularization method.

4.2.4 Video representation and classification

We use our trained network for frame-level feature and emotion intensity extraction in video emotion recognition. As with [100], each video frame is passed to our trained network, the activations from the $pool_5$ layer are extracted as the frame-level feature, and the average value of the predicted intensity map is calculated as emotion intensity for the input frame. As we use the ResNet as the backbone network, the dimension of the $pool_5$ layer is 2048. We apply $L2$ normalization to frame-level features. Let $\boldsymbol{f}_i$ and w_i denote the extracted frame-level feature and emotion intensity for the i-th frame, respectively, the video-level feature for a video clip is represented as follows:

$$\boldsymbol{F} = \sum_{i=1}^{\bar{N}} w_i \boldsymbol{f}_i, \tag{4.7}$$

where $\bar{N}$ denotes the number of frames in the clip. The obtained video-level features are further $L2$ normalized. The dimension of video-level features is the same as frame-level features. We train an SVM with the linear kernel on video-level features to classify test videos into different emotion categories.

4.2.5 Training details

We train our network on the WEBEmo dataset [71]. This dataset totally contains 268,000 images, which were selected from initially 300,000 weakly labeled images retrieved from the Web. Currently, it is the largest public dataset for image emotion recognition tasks. This dataset was labeled at three levels according to Parrotts hierarchical model of emotions [14]. In this work, we use labeling method at level three which categorizes the images into 24 emotions. 75% of the images are randomly selected for training, and the rest are for testing. The train/test split is provided with this dataset. The authors showed that deep models learned on this dataset could help minimize the effect of dataset bias for sentiment analysis.

The proposed network is implemented in Pytorch [105]. All training images were resized with the size of the shorter side equal to 256 while maintaining the original aspect ratio. A 224×224 image was randomly cropped from original image or its horizontal flip

as input data to networks. Each channel of input data was normalized to have zero mean and unit variance. The values of λ_0, λ_1, λ_2, λ_3 are set to 2, 1, 10, and 1, respectively. The values of weight decay and momentum are set to 0.0001 and 0.9, respectively. We train our network using stochastic gradient descent (SGD) for 90 epochs with a mini-batch of size 128. The learning rate starts from 0.1 and is divided by 10 at epoch 30 and 60.

4.3 Experiments

4.3.1 Experimental Setup

To evaluate the performance of the our approach, experiments were conducted on two benchmark datasets, VideoEmotion-8 and Ekman-6.

Implementation Details In our experiments, all selected video frames were resized with the size of the shorter side equal to 256 while maintaining the original aspect ratio. For each selected frame, we cropped 10 regions of the size of 224×224 (four corners and one center, and their horizontal flip). The 10 crops were fed to our trained network, and activations of the $pool_5$ layer were extracted. The average of the 10 activations was calculated as the frame-level representation. All frame-level features were $L2$ normalized. A multi-class SVM was trained using the LibSVM toolbox [106] with the radial basis function kernel on video-level features for classification.

4.3.2 Results on VideoEmotion-8

The overall recognition accuracy achieved by the proposed method is shown in Table 4.1. We see from this table that using the proposed weighted sum pooling achieve a 2.89% performance gain as compared to the average pooling method, improving the overall recognition performance from 49.98% to 52.87%. This demonstrates the effectiveness of utilizing frame-level emotion intensity for improving performance.

We also compared the performance of features extracted from different deep neural networks. As with [100], we used AlexNet, VGG-16, VGG-19, and ResNet-152, which were pretrained on ImageNet. The activations of the fc_7 layer were extracted from AlexNet, VGG-16, and VGG-19 as frame-level features, and the activations of the $pool_5$ layer were extracted from ResNet-152 as frame-level features. The average pooling was

Feature type	Overall accuracy
AlexNet (fc_7)	43.0
VGG-16 (fc_7)	44.8
VGG-19 (fc_7)	44.0
ResNet-152 ($pool_5$)	45.5
Our network ($pool_5$) + Avg. pooling	**49.98**
Our network ($pool_5$) + Wgt. sum pooling	**52.87**

Table 4.1 : Performance of the proposed approach and comparison with features extracted from other deep networks on VideoEmotion-8.

used to generate video-level features. We observe from Table 4.1 that the $pool_5$ features extracted from ResNet-152 achieve the best recognition accuracy of 45.5% among the four types of features. This is 4.48% lower than using the $pool_5$ features extracted from the proposed network pretrained on WEBEmo.

The confusion matrix of the recognition accuracy achieved by our method is shown in Figure 4.3. We find that most categories are confused with "surprise". This might be because the "surprise" category has the largest number of training samples, which leads the classifier to predict in favour of this category. However, our method performs badly for the "anticipation" and "trust" categories. The performance is relatively well for the rest of categories.

Comparison with previous work To demonstrate the superiority of the proposed approach, we compared the performance of our method with the recent work on VideoEmotion-8. The comparison results are shown in Table 4.2.

The results show that we achieve better performance than previous work using a single-type feature or combination of multi-type feature. The proposed method outperforms Sentibank and EMDBM by over 12.0%. Similar as our approach, the image transfer

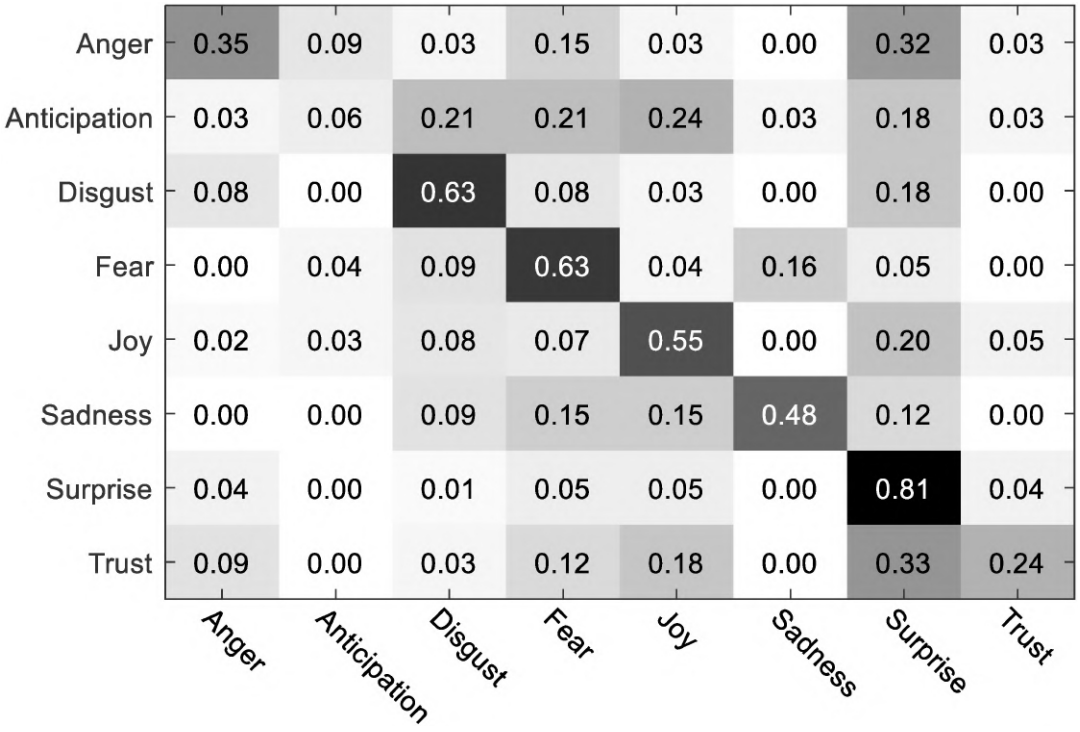


Figure 4.3 : Confusion matrix on VideoEmotion-8 using the proposed approach.

Method	Overall accuracy(%)
SentiBank [64]	35.5
EMDBM [5]	40.4
ITE [54]	44.7
Kernelized features [100]	49.7
Vis.+Aud.+Att. [48]	46.1
Context Fusion Net [51]	50.4
Vis.+Aud.+Att. +EMDBM [5]	51.1
Our approach	**52.87**

Table 4.2 : Comparison with previous work on VideoEmotion-8.

encoding (ITE) method learns video representations from an auxiliary image dataset. Our approach achieves 8.17% higher performance than the ITE method. In previous work, kernelized features achieve the best recognition performance using a single-type feature. Our approach also outperforms kernelized features by 3.17%. This demonstrates the advantage of our method for emotion recognition in videos. Previous studies have employed combining multi-type features to improve performance. In [48], low-level visual, audio, and attribute features were combined to improve video emotion recognition performance. In [5], Pang *et al.* proposed an approach to learn joint representations on multimodal inputs, including visual, audio, and textual modalities, using the deep Bolzmann machine [68] for emotion prediction. Compared with [48] and [5], our method achieves 6.77% and 1.77% improvements, respectively. When compared with [51], in which features were extracted from multiple pre-trainied CNNs, and were further fused by a neural network for generating video-level representations, the proposed method yields an improvement of 2.47%.

We further compared the performance of our method with the $pool_5$ features extracted ResNet-152 and kernelized features for each emotion category. The comparison results are shown in Figure 4.4. We observe that our method improves the performance for "disgust", "fear", "joy", "sadness", and "surprise" compared to ResNet($pool_5$) features and kernelized features, with the largest improvement for 'disgust'. For the emotions of anger and anticipation, we achieve lower performance than ResNet($pool_5$) features and kernelized features.

4.3.3 Results on Ekman-6

The experimental results yielded by the proposed method on Ekman-6 is shown in Table 4.3. As with on VideoEmotion-8, weighted sum pooling achieves higher performance than average pooling using features extracted from our network, improving the overall recognition accuracy from 54.55% to 56.50%. We compared the performance of our approach with using four types of deep features, *i.e.,* AlexNet, VGG-16, VGG-19, and ResNet-152, which were pretrained on ImageNet. As with on VideoEmotion-8, the $pool_5$ features extracted from ResNet-152 achieved the best performance among the types of features. This might be because ResNet are superior compared to AlexNet and

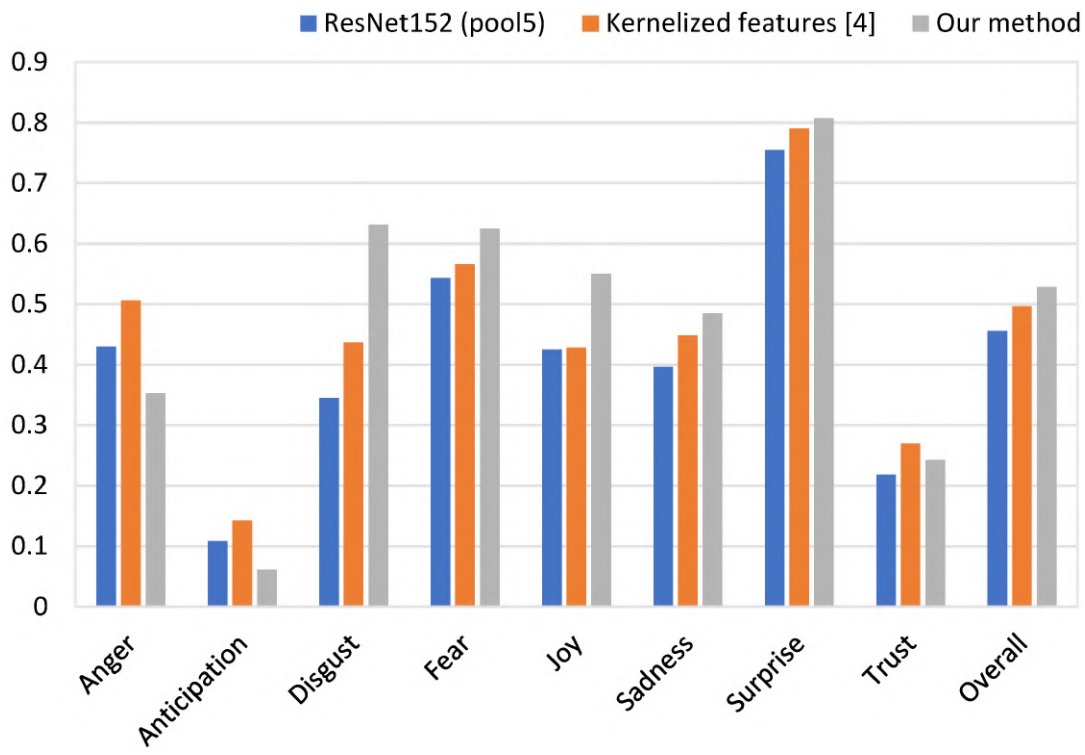


Figure 4.4 : Recognition accuracy for each emotion category on VideoEmotion-8.

VGGNets, and thus features extracted from ResNet are more discriminative than those extracted from AlexNet and VGGNets. Compared with ResNet-152($pool_5$) features, the features extracted from our pretrained network with weighted sum pooling achieves an 6.5% performance gain .

The confusion matrix of the recognition accuracy yielded by the proposed approach is shown in Figure 4.5. We see from this figure that most prediction results are confused with the "surprise" category, which is the same on the VideoEmotion-8 dataset. Our method achieves the best performance of 63.0% for the emotion of joy. For all emotion categories other than "anger", the proposed method achieves over 50.0% prediction accuracy. The deviation of the distribution of prediction accuracies for the six categories on Ekman-6 is small compared to on VideEmotion-8.

Comparison with previous work We compared the performance of our method with the recent work on Ekman-6 to show the advantage of our method. The comparison results are shown in Table 4.4.

Feature type	CNN
AlexNet (fc_7)	48.3
VGG-16 (fc_7)	49.6
VGG-19 (fc_7)	49.8
ResNet-152 ($pool_5$)	50.0
Our network ($pool_5$) + Avg. pooling	**54.55**
Our network ($pool_5$) + Wgt. sum pooling	**56.50**

Table 4.3 : Performance of the proposed approach and comparison with features extracted from other deep networks on Ekman-6.

Method	Overall accuracy (%)
ITE [54]	50.4
Context Fusion Net [51]	51.8
Kernel features [100]	**54.4**
Our approach	**56.50**

Table 4.4 : Comparison with previous work on Ekman-6.

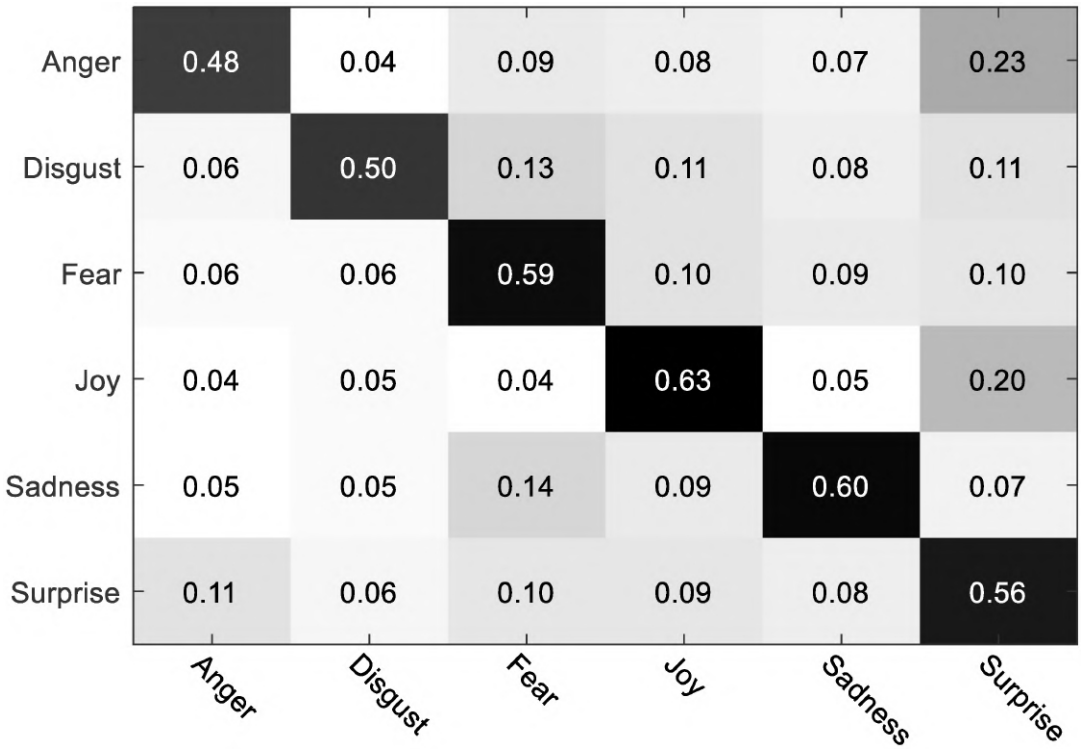


Figure 4.5 : Confusion matrix on Ekman-6 using the proposed approach.

The proposed approach outperforms the ITE method by 6.10%. Compared with Context Fusion Net, our method achieves an improvement of 4.7%. In previous work, kernelized features have achieved the state-of-the-art performance on this dataset, wherein they obtain 54.4% overall recognition accuracy. Our method achieves 56.50% recognition accuracy, yielding a 2.10% improvement compared with kernelized features . This shows the effectiveness of our method for performance improvement in recognition of emotions in videos.

Furthermore, we compared the performance of the proposed method with ResNet($pool_5$) features and kernelized features for each emotion category in Ekman-6. The comparison results in shown in Figure 4.6. It can be seen from Figure 4.6 that our method improves the prediction accuracy for “disgust”, “fear”, “joy”, and “sadness”. For the other two emotion categories, our method performs poorly compared to kernelized features. Our method achieves significant improvements for the emotions of joy and sadness, yielding over 15% improvement.

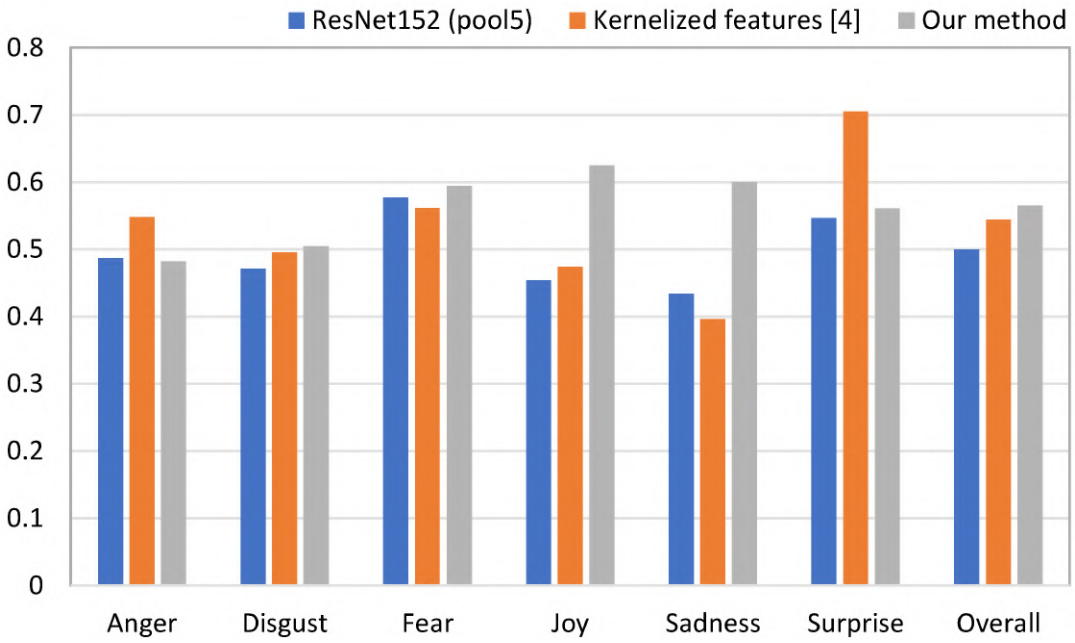


Figure 4.6 : Recognition accuracy for each emotion category on Ekman-6.

Qualitative analysis We conducted qualitative analysis on the Ekman-6 dataset. Examples of predicted emotion intensity maps for video frames in Ekman-6 is shown in Figure 4.7. While the proposed neural network is trained on WEBEmo, it can be seen from Figure 4.7 that the proposed network has good generalization performance for emotion intensity prediction on the Ekman-6 dataset. Due to the bias between WEBEmo and the video frames, there are many predicted emotion intensity maps that do not accurately highlighted discriminative emotion regions for video frames. However, using the average value of predicted intensity maps is help to improve recognition performance compared to using the simple average pooling method.

4.4 Conclusions

In this chapter, we have presented a weighted sum pooling method for video emotion representation. A deep neural network is developed for simultaneously image classification and emotion intensity prediction. The proposed network can be trained in an end-to-end manner. The proposed network is first trained on the WEBEmo dataset, which is currently the largest image emotion dataset. Then, we use it for frame-level features and

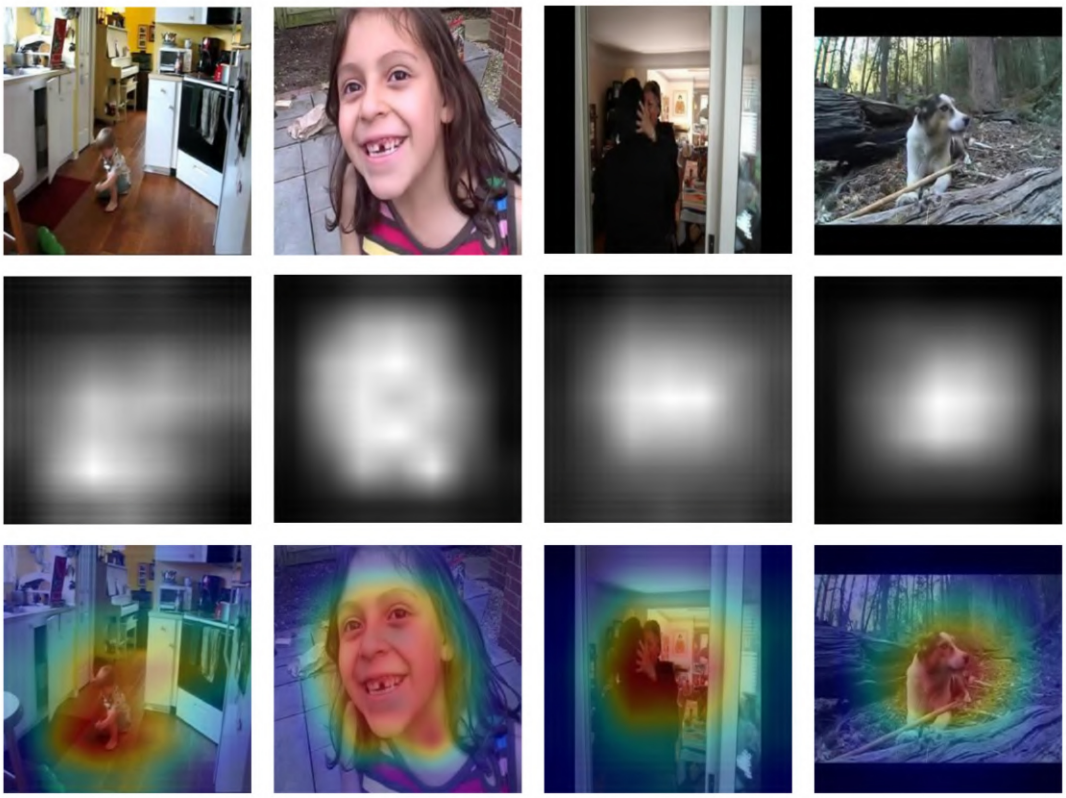


Figure 4.7 : Examples of predicted emotion intensity maps for video frames on Ekman-6.

emotion intensity maps extraction. The average values of the predicted intensity maps are calculated as weights in our weighted sum pooling to generate video-level features. The effectiveness of our approach is evaluated on two benchmark datasets.

In this work, frame-level emotion intensities are used as weights for generating video-level features. In the future work, we would explore using emotion intensity to select discriminative video frames and adapting the proposed neural network to the selected video frame domain.

Chapter 5

Weakly Supervised Emotion Intensity Prediction for Recognition of Emotions in Images

5.1 Introduction

Image emotion recognition aims to predict the invoked emotion in humans after they see an image. This is different from cognitive content analysis tasks such object detection, scene classification [107] and human action recognition, which aims to analyze visual content at semantic level. The main challenge of visual emotion recognition is that emotions are much high level of abstraction over visual semantics. The images that conveyed a certain category of emotion can be taken from very difference scenes with various objects and/or different people. The large intra-class variation makes it extremely difficult for visual emotion inference.

Early studies deal with this challenge by designing discriminative low-level and high-level hand-crafted image features or by combining multi-type features or multimodal features such as visual and textual features [5, 47]. Over the past few years, deep neural networks have become a dominant approach in a variety of computer vision tasks. Deep neural networks can automatically learn much robust features from images. It is well known that deep learning approaches thrive on huge amounts annotated training data, *e.g.,* ImageNet-2012 which . Recently, with the availability of large scale image emotion datasets [59, 71], researchers started to apply deep learning methods for image emotion recognition. The work of [7, 8] has shown that deep learning methods exhibit much superior performance for image emotion recogniton compared with hand-crafted features. However, as compared to on other computer vision tasks such as image classification and scene understanding, the performance of state-of-the-art deep neural networks, such as ResNet [27] and ResNeXt [29], is poor on image emotion recognition.

Most existing datasets for image emotion recognition are annotated at only image-

Figure 5.1 : Sample images and corresponding emotion intensity maps synthesized with the original image. As shown in the second row, emotion intensity maps highlight discriminative regions that invoke an emotion.

level. Image-level labels indicate the presence of the dominant emotion somewhere in the image, while the other parts may convey a neutral emotion or other emotions. Image-level annotations are rather weak. Compared to whole images, local image regions may contain discriminative information for emotion inference. With region information, deep networks can learn more robust features. The recent work of [7,8] has shown that employing region information helps to improve emotion recognition performance. A limitation of these methods is that they need bounding box annotations, which require exhaustive work to obtain, or a computationally extensive procedure to discover emotion regions. Intuitively, emotion intensity maps provide more emotional information than image regions (see Figure 5.1). There has been work [108] employing emotion intensity to improve visual saliency computing. Unlike region based methods for image emotion recognition, we go one step further by utilizing emotion intensity. To the best of our knowledge, none of the existing methods have considered employing emotion intensities for visual emotion recognition. Manually labelling emotion intensity maps would require huge amounts of work, especially when considering the subjectivity of emotions. In this work, we tackle

the problem through weakly supervised emotion intensity learning.

There has been work to extract saliency maps for images from a classification convolutional network trained using image labels. In [109], Simonyan *et al.* proposed a method for computing image-specific class saliency maps using a single back-propagation pass with a pretrained classification convolutional network. This is the first work to obtain attention maps for localizing objects of interest using only image-level annotations. The work of Zhou *et al.* [10] showed that the class activation mapping (CAM) technique can localize discriminative image regions that are used by the deep network to identify their category. The CAM technique has been used to provide cues for weakly object detection [110] and weakly supervised semantic segmentation [111], where promising results have been obtained. Unlike convolutional network derived approaches, we hypothesize that intensity maps can be directly inferred from input images. In this work, we aim to employ emotion intensity to improve image emotion recognition performance. Based on our hypothesis, we develop an end-to-end deep neural network that simultaneously output emotion intensity map and emotion recognition result.

An overview of the proposed network is shown in Figure 5.2. The proposed network consists of two streams: an emotion intensity prediction stream and a classification stream. The intensity prediction stream is built on top of the feature pyramid network (FPN) [101], which can extract multi-level features from the image. The CAM technique is used to generate pseudo intensity maps to guide the proposed network for emotion intensity learning. Three loss functions are employed in this work to improve intensity prediction performance. The predicted emotion intensity maps are utilized by the classification stream final emotion recognition. The proposed network is trained with a multi-task loss function The two streams are trained cooperatively with each other to improve overall performance.

The main contributions of this chapter can be summarized as follows.

- This is the first work on image emotion recognition by utilizing emotion intensity. We propose an end-to-end deep neural network which consists of an intensity prediction stream and a classification stream. The predicted intensity maps high-

light discriminative emotional regions which are used by the classification stream for final emotion recognition. The proposed network can learn more robust features compared to region-based approaches.

- We evaluate the proposed network for image emotion recognition and sentiment analysis. Extensive experimental results show that the proposed network is effective to improve recognition performance.

5.2 Methodology

The goal of this work is to improve image emotion recognition performance employing emotion intensity. Unlike CNN-derived approaches, our method intends to predict emotion intensity directly from the input image. An overview of the proposed network is shown in Figure 5.2. The proposed network consists of two streams: an emotion intensity prediction stream and an emotion classification stream. The predicted emotion intensity map is integrated to the classification stream for final emotion recognition. The proposed network can be trained end-to-end.

As there are only image-level annotations in most datasets, we train our network by weakly supervised learning with pseudo emotion intensity maps. In the following subsections, we first introduce the method for generating pseudo intensity maps. Then, we present the details of the network architecture and the loss functions for training the network.

5.2.1 Pseudo intensity map generation

As introduced in Section 4.1, saliency maps can be derived from a pretrained CNN. In this work, we use the saliency maps derived from a CNN as pseudo emotion intensity maps to guide the proposed network for emotion intensity learning.

In particular, we follow the work of Zhou *et al.* [10] to generate intensity maps with the CAM technique. The deep residual network (ResNet) [27] is used as the backbone network. The ResNet consists of one convocational layer and four convolutional blocks, each convolutional block contains a number of bottleneck layers. This is followed by a global average pooling layer and a fully connected layer with softmax. In our experi-

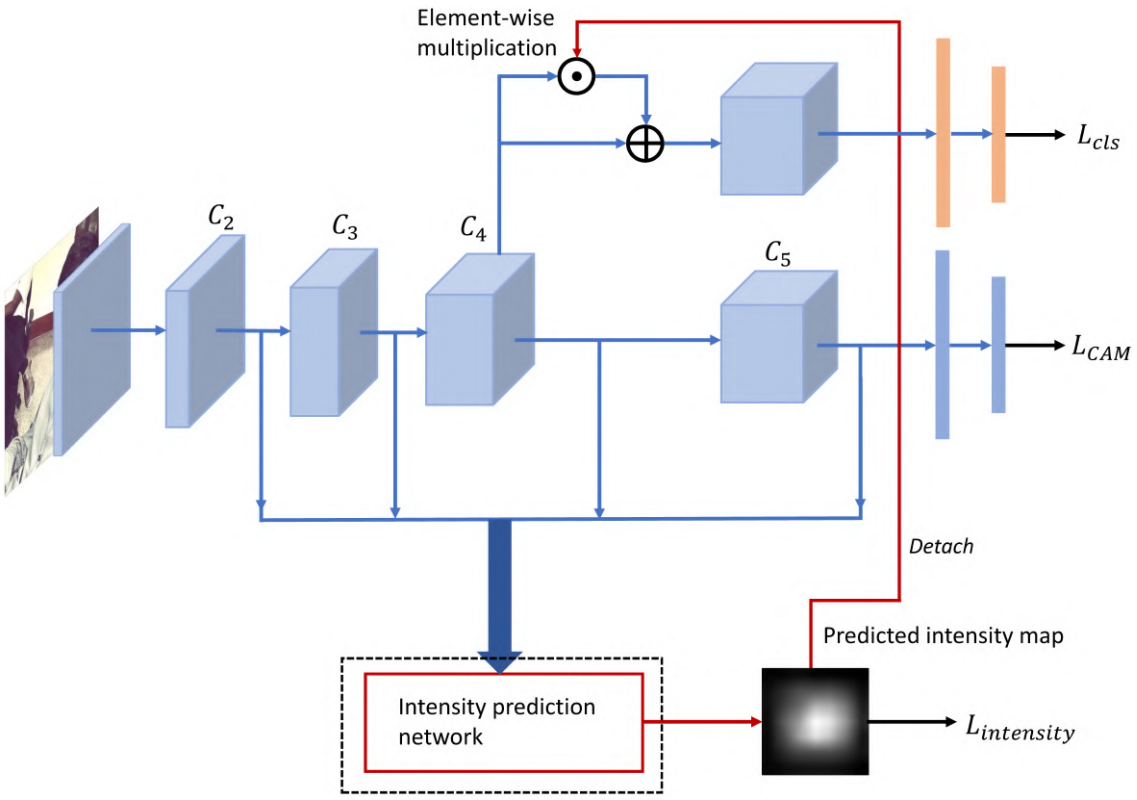


Figure 5.2 : An overview of the proposed end-to-end network architecture for image emotion recognition. This network consists of an emotion intensity prediction stream and a classification stream. The predicted intensity maps are integrated to the classification stream for final emotion recognition. The proposed network is trained with a multi-task loss function. The two streams are trained cooperatively with each other to improve overall performance.

ments, we explored two ResNet structures: ResNet-50 and ResNet-101. As with [10], we ignore the bias term in the last layer as it has trivial impact on classification performance. The global average pooling layer outputs the spatial average of the feature maps produced by the last convolutional layer. The CAM technique identifies the discriminative region by projecting back the weights of the fully connected layers on to the convolutional feature maps. The class activation map is calculated as the weighted linear summation of the presence of the visual patterns at different spatial locations. Class activation maps for a category of interest depicts the saliency region used by the network to identify that cate-

gory. Let the class activation map for class u be denoted M_u, then it can be represented as follows:

$$M_u(x, y) = \sum_k w_k^u f_k(x, y), \tag{5.1}$$

where $f_k(x, y)$ denotes the activation of k-th feature map produced by the last convolutional layer at the spatial local (x, y), and w_k^u denotes the weight corresponding to class u for the unit k of the last fully connected layer.

The values of the obtained M_u are rescaled to 0-255 as the emotion intensity map. The value of an emotion intensity map represents to what degree this area represents an emotion. Emotion intensity maps provide discriminative local information that can be used to improve emotion recognition performance. The size of the generated intensity maps is the same as the feature maps produced by the last convolutional layer. By upsampling the obtained intensity maps to the desired size, we can use them to guide the proposed network network for supervised emotion intensity map learning.

5.2.2 The network architecture

The emotion intensity prediction stream

Our network for emotion intensity map learning is built on top of the FPN (see Figure 5.3). With FPN, the network can learn multi-scale semantically strong features. The FPN architecture consists of a bottom-up pathway, a top-down pathway, and lateral connections. The bottom-up pathway is the backbone network which computes multi-scale feature maps by feedforward computation. Feature maps generated by up layers have low resolution. The feature maps produced by the last bottleneck layer of the four convolutional blocks are selected to be in a feature pyramid. They will be connected to the top-down pathway by lateral connection. Let the selected feature maps be denoted by C_2, C_3, C_4, and C_5, respectively (see Figure 5.3). Then, the feature pyramid can be represented as $\{C_2, C_3, C_4, C_5\}$. Following the work of Lin [101], the feature maps generated by the first convolutional layer are not included to the feature pyramid due to their large memory cost.

In the top-down pathway, coarse resolution feature maps are upsampled to be of high spatial size. The lateral connections merge feature maps of the same spatial size from the

bottom-up pathway and the top-down pathway. In particular, we apply the upsampling operation with a factor of 2 (the bilinear interpolation is used in this work) to coarse resolution feature maps. The upsampled feature maps is then merged with the corresponding feature maps in the feature pyramid $\{C_2, C_3, C_4, C_5\}$ by element-wise addition. 1×1 convolutions with the output channel number equal to 256 are applied to the feature maps in the pyramid to reduce channel dimensions before addition. Let the feature maps obtained after addition be denoted by P_i, then it can be represented as follows:

$$P_i = Conv_{1\times1}(C_i) + Up_{2\times}(P_{i+1}), i = 2, 3, 4. \tag{5.2}$$

P_5 is directly produced by lateral convolutions. After the top-down pathway, we obtain the set of feature maps $\{P_2, P_3, P_4, P_5\}$, each with 256 channels, corresponding to $\{C_2, C_3, C_4, C_5\}$, respectively. Then, the feature maps in $\{P_2, P_3, P_4, P_5\}$ are connected to a stack of two 3×3 convolutional layers with the output channel number equal to 128. After that, we get the set of feature maps $\{Q_2, Q_3, Q_4, Q_5\}$, each with 128 channels, corresponding to $\{P_2, P_3, P_4, P_5\}$. The feature maps in $\{Q_2, Q_3, Q_4, Q_5\}$ are concatenated to generated 512 channels of feature maps. Because the size of the these feature maps is different, they are upsampled to be of the same spatial size before concatenation. This is further followed by two consecutive 3×3 convolutional layers, which end up with a sigmoid layer to produce emotion intensity maps. We do not use nonlinear transformation in the top-down pathway. The size of outputs is $1/4$ of original input size.

5.2.3 The loss functions

To improve intensity map prediction performance, three loss functions, *i.e.,* root mean square error in log space (RMSEL) [102], gradient loss, and surface normal loss, are used in our experiments. The details of the three loss functions are introduced as follows.

1. The RMSEL loss This loss function was originally introduced for depth prediction [112]. The RMSEL loss empirically achieves better performance than root mean square error (RMSE), $L1$-norm loss, and $L2$-norm loss for emotion intensity prediction. The RMSEL loss function is defined as follows:

$$L_{RMSEL} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sqrt{(log(p_{ij}) - log(g_{ij}))^2}, \tag{5.3}$$

where p_{ij} and g_{ij} denote the values of the predicted intensity map and of the pseudo intensity map generated by the CAM technique, respectively, at the spatial location (i, j) and N represents the height and width of the predicted intensity map.

2. Gradient loss The gradient of an intensity map $\boldsymbol{f}$ is given as follows:

$$\nabla \boldsymbol{f} = [\frac{\partial \boldsymbol{f}}{\partial x}, \frac{\partial \boldsymbol{f}}{\partial y}], \tag{5.4}$$

where $\frac{\partial \boldsymbol{f}}{\partial x}$ and $\frac{\partial \boldsymbol{f}}{\partial y}$ denote the partial derivative of the intensity map with respective to the x direction and the y direction, respectively. In this work, the gradient of intensity maps is obtained by applying two predefined Sobel filters [103] (see the Appendix 5.A section for more details). The gradient loss is represented as the L_1 norm of the difference between the gradients of the predicted intensity map and of the pseudo intensity map, which is given as follows:

$$L_{gradient} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i=1}^{N} \left\| \nabla d_{ij} - \nabla p_{ij} \right\|_1, \tag{5.5}$$

where $\|\cdot\|$ denotes the L_1-norm, and ∇d_{ij} and ∇p_{ij} denote the gradients of the predicted intensity map and of the pseudo intensity map generated by the CAM method computed at the spatial location (i, j), respectively. The gradient loss penalizes errors around edges [104].

3. Surface normal loss The surface normal loss [104] measures the difference between the normal to the surface of the estimated intensity map and the normal to the surface of the pseudo intensity map. Specifically, the surface normal loss is defined as follows:

$$L_{normal} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i=1}^{N} (1 - \frac{\langle n_{ij}^d, n_{ij}^g \rangle}{\sqrt{\langle n_{ij}^d, n_{ij}^d \rangle} \sqrt{\langle n_{ij}^g, n_{ij}^g \rangle}}), \tag{5.6}$$

where and $\langle \cdot, \cdot \rangle$ denotes inner product of two vectors, which measure the angle between two surface normals, $n_{ij}^d = [-\nabla_x(d_{ij}), -\nabla_y(d_{ij}), 1]$ and $n_{ij}^g = [-\nabla_x(g_{ij}), -\nabla_y(g_{ij}), 1]$ denotes the surface normals of the predicted intensity map and of the pseudo intensity map generated by the CAM method, respectively, at the spatial location (i, j).

The overall emotion intensity loss function is defined as follows:

$$L_{intensity} = \lambda_1 L_{RMSEL} + \lambda_2 L_{gradient} + \lambda_3 L_{normal}. \tag{5.7}$$

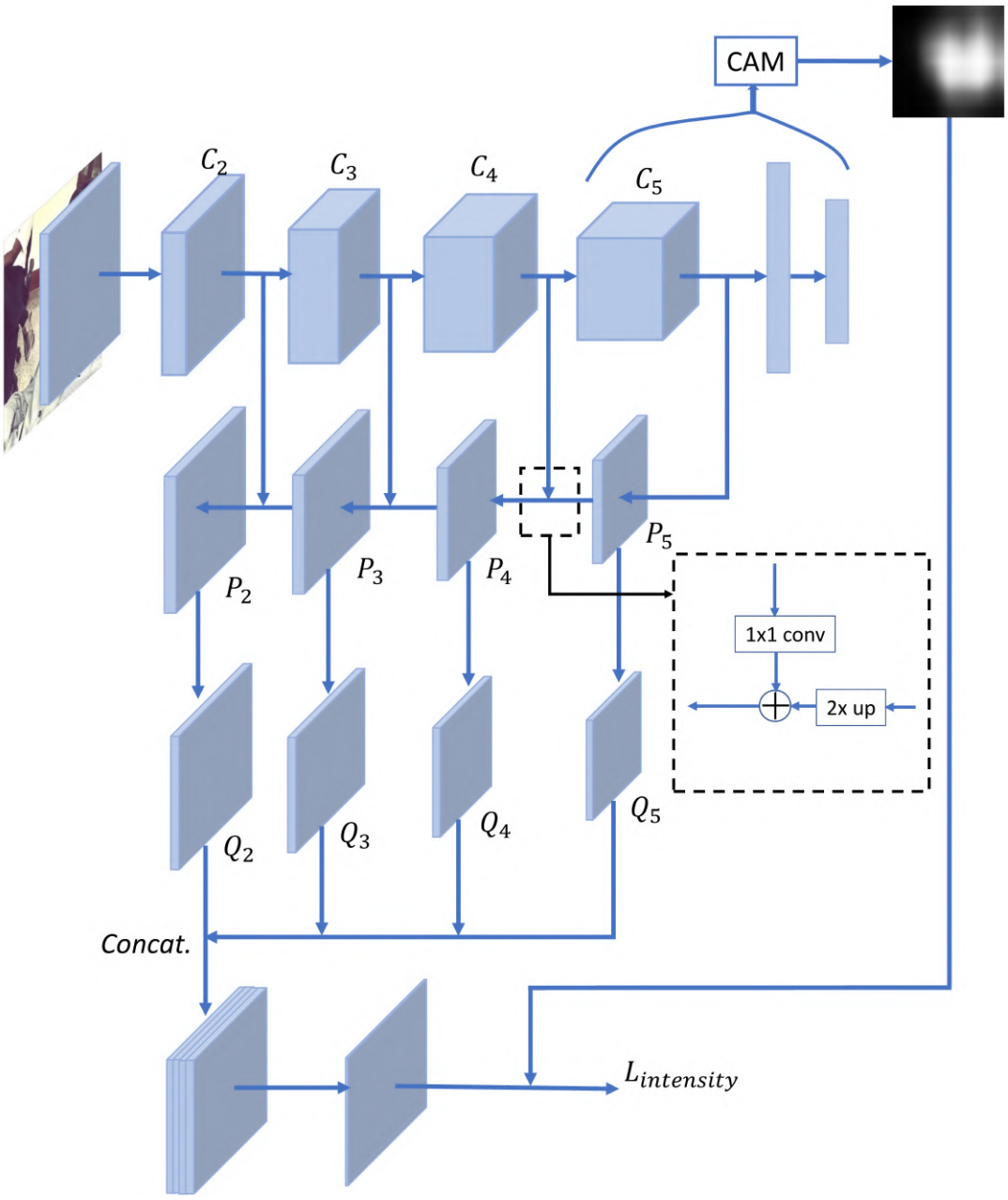


Figure 5.3 : The emotion intensity prediction subnetwork. The subnetwork is built on top of the FPN. The CAM technique is used to generate pseudo intensity maps to guide the subnetwork for emotion intensity learning.

The normal loss and the gradient loss help to refine predicted intensity maps. The impact of the three loss functions on recognition performance will be discussed in the experiment section.

Improving classification performance with intensity maps

The predicted intensity map is integrated to the classification stream for final emotion classification. We multiply the $C3$ feature maps by the intensity map using element-wise multiplication. The intensity map is first interpolated to be of the same size as $C3$ before element-wise multiplication. The obtained feature maps are then fused together with the original $C3$ features by element-wise addition:

$$\overline{C3} = I \odot C3 + C3, \tag{5.8}$$

where $\odot$ denotes element-wise multiplication and I denotes the predicted intensity map after interpolation. This could lead the network to pay more attention to discriminative regions. The $\overline{C3}$ feature maps are taken as input to a subnetwork. The subnetwork consists of a convolutional block and a global average, followed by a fully connected layer. The parameters of the subnetwork are shared with the subnetwork for generating pseudo intensity maps.

Finally, the whole network is trained with the following multi-task loss:

$$\begin{aligned} L_C &= L_{intensity} + \lambda_4 L_{CAM} + \lambda_5 L_{cls} \\ &= \lambda_1 L_{RMSEL} + \lambda_2 L_{normal} + \lambda_3 L_{gradient} \\ &\quad + \lambda_4 L_{CAM} + \lambda_5 L_{cls}. \end{aligned} \tag{5.9}$$

As the final classification stream and the subnetwork for generating pseudo intensity maps all output prediction results, two classification losses, *i.e.,* L_{cls} and L_{CAM}, are used in the multi-task loss. We use the cross entropy loss for the two classification losses.

The proposed network can be trained end-to-end. The two network streams work cooperatively with each other. The emotion intensity prediction stream helps to improve classification performance. The quality of predicted pseudo intensity maps is refined as the classification accuracy increases. This in turn helps to improve emotion intensity prediction performance.

5.3 Experiments

We conducted extensive experiments on three benchmark datasets to show the proposed network is effective for image emotion recognition. We further evaluated our network on image sentiment analysis.

5.3.1 Experimental setup

Datasets

The experiments were conducted on three benchmark datasets.

Emotion-6 [71] This dataset consists of 8350 images, which were selected from initially 150K images crawled from Google and Flickre. The images in this dataset were labeled with six emotion categories ('anger', 'fear', 'joy', 'love', 'sadness', and 'surprise') according Ekman's research on basic human emotions [113]. Following the work of Panda [71], 80% of the images were randomly selected for training, and the rest were used for testing.

The FI-8 dataset [59] This dataset was collected from Flickr and Instagram. There are totally 23,308 images labelled with eight emotion categories, which are defined by the psychological study of Mikels [114], by Amazon Mechanical Turk workers. As some images are no longer exist on the Internet, only 23,308 images were crawled for our experiments. The dataset was split 80%, 5%, and 15% for training, validation, and testing, respectively. In this work, the model for reporting performance was trained on the combination of training and validation sets.

WEBEmo [71] This dataset consists of approximately 268,000 images, which were retrieved from the Web. Currently, this is the largest dataset publicly available for image emotion recognition. The images in this dataset were labeled with 24 emotion categories and also labeled with six categories according to Parrotts hierarchical model of emotions [14], denoted as WEBEmo-24 and WEBEmo-6, respectively, in this thesis. The dataset is split 80%, and 20% for training and testing, respectively.

Model	Recognition accuracy (%)
AlexNet	51.35
DensNet-169	57.17
ResNet-50 (vanilla) [71]	54.99
Ours (ResNet-50)	**59.75**
ResNet-101 (vanilla)	56.69
Ours (ResNet-101)	**60.41**

Table 5.1 : Recognition accuracy (%) of the proposed network on Emotion-6.

Implementation details

We implemented the proposed network in Pytorch [105]. The backbone network was initialized using weights pretrrained on ImageNet [115]. The values of λ_1, λ_2, and λ_3 were set to 1, 10, and 1, respectively, for ResNet-50, and the values of λ_1, λ_2, and λ_3 were set to 0.5, 5, and 0.5, respectively, for ResNet-101. The values of λ_4 and λ_5 were all set to 1. The values of weight decay and momentum were set to 0.001 and 0.9, respectively. We trained the models using the stochastic gradient descent (SGD) algorithm for 90 epochs with a mini-batch of 128 samples on four GPUs (32 images per GPU). The learning rate was initialized to 0.001, and was divided by 10 at epoch 30 and 60. All training images were resized with the size of the shorter side equal to 256 while maintaining the original aspect ratio. A 224×224 image was randomly cropped from the original image or its horizontal flip as input data to the network. Each channel of input data was normalized to have zero mean and unit variance.

At test time, the network made a prediction by cropping 10 regions of the size of 224×224 (four corners and one center, and their horizontal flip) from a test image, and averaging the predictions made by the networks softmax layer on the ten patches.

5.3.2 Results on Emotion-6

The overall recognition accuracy on Emotion-6 is shown in Table 5.1. From this table, we see that our method achieves 4.76% and 3.72% performance improvements

Loss function	Recognition accuracy (%)
L_{cls}	54.99
L_{cls} + $\lambda_1 L_{RMSEL}$	57.12
L_{cls} + $\lambda_1 L_{RMSEL}$ + $\lambda_2 L_{norm.}$	58.61
L_{cls} + $\lambda_1 L_{RMSEL}$ + $\lambda_3 L_{grad.}$	58.48
L_{cls} + $\lambda_1 L_{RMSEL}$ + $\lambda_2 L_{norm.}$ + $\lambda_3 L_{grad.}$	59.75

Table 5.2 : Impact of loss function on performance on Emotion-6. The ResNet-50 was used as the backbone network in the experiments.

for ResNet-50 and ResNet-101, respectively. The proposed network on ResNet-101 also outperforms AlexNet and DenseNet-169 by 9.06% and 3.24%, respectively. This shows the effectiveness of the proposed method to improve recognition accuracy.

The confusion matrix on Ekman-6 using the proposed network based on ResNet-101 is shown in Figure 5.5. We observe from this figure that most categories are confused with “joy” and “sadness”. This might be because the two categories have more training samples than the other categories, which leads the classifier to predict in favour of the two categories. Our network performs badly for the emotion of fear. Figure 5.4 shows the examples of predicted intensity maps by the proposed network and pseudo intensity maps generated by the CAM technique. It can be seen from Figure 5.4 the predicted intensity maps are good approximations to the CAM intensity maps. While the proposed network performs poorly in predicting intensity maps in some cases, employing predicted intensity maps helps to improve overall recognition accuracy.

We conducted an ablation study to show the impact of the emotion intensity loss function on recognition performance. We used the the proposed network based on ResNet-50. The results are shown in Table 5.2. It can be seen from Table 5.2 that using only the

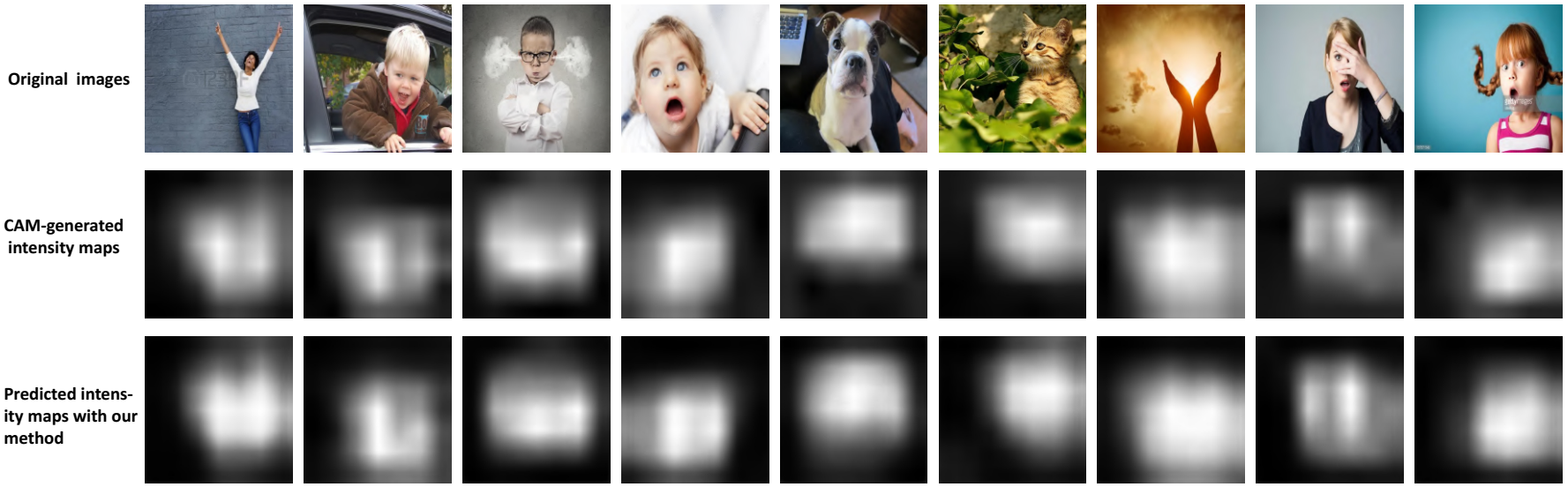


Figure 5.4 : Examples of emotion intensity maps generated by CAM and predicted with the proposed network.

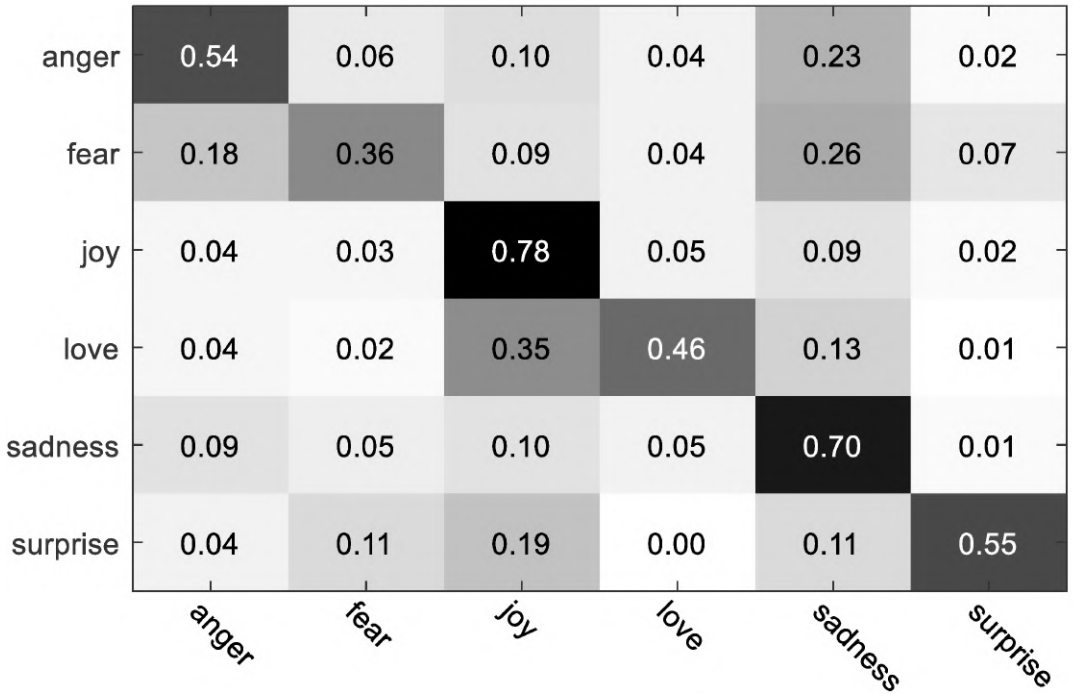


Figure 5.5 : The confusion matrix on Emotion-6 using the proposed network based on ResNet-101.

RMSEL loss improves the overall recognition accuracy from 54.99% to 57.12%. Further adding the gradient loss or the surface normal loss yields approximately 1.40% performance improvement. Using the combination of the three losses achieves the final accuracy of 59.75%. This shows that the gradient loss and the surface normal loss are complimentary with each other to improve recognition accuracy.

5.3.3 Results on FI-8

The experimental results on FI-8 is shown in Table 5.3. Our networks based on ResNet-50 and on ResNet-101 achieve 74.84% and 75.91% overall recognition accuracy, respectively. Figure 5.6 shows the confusion matrix on FI-8 using the proposed network based on ResNet-101. We see from Figure 5.6 that most emotion categories are confused with the “contentment” category. the proposed network performs poorly for “anger” and “fear”. Our networks achieves at least 70.0% recognition accuracy for the rest of emotion categories.

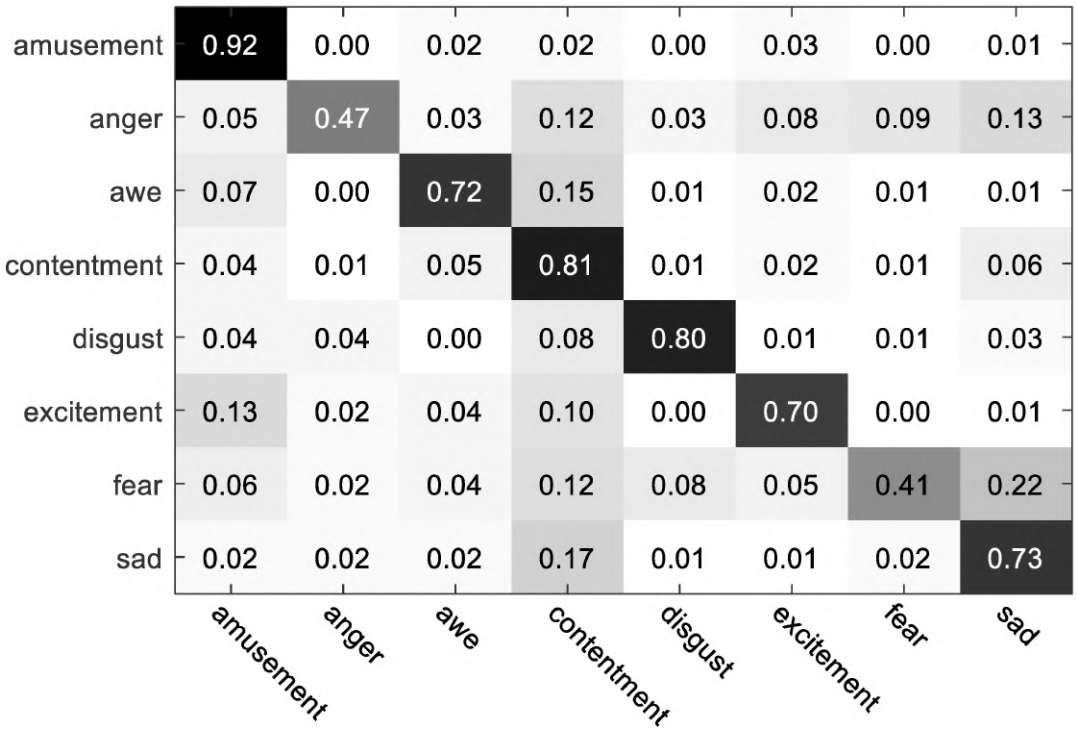


Figure 5.6 : The confusion matrix on FI-8 using the proposed network based on ResNet-101.

We compared our method with the latest work on the FI-8 dataset (see Table 5.3). Our network outperforms Sentibank by over 30.0%. We also achieve better recognition performance than other deep-based approaches such as AlexNet and ResNet-101. In the work of Rao [8], the multi-level region-based network was used for image emotion classification. The proposed network outperforms Rao's by 0.45%. While Rao's method can be trained end-to-end, their network needs to be pretrained on a dataset with region annotations. Whereas our network can be directly trained using only frame-level labels in an end-to-end manner. This demonstrates the advantage of using weakly supervised emotion intensity prediction compared to detecting discriminative emotion regions.

5.3.4 Results on WEBEmo

We conducted experiments on both WEBEmo-6 and WEBEmo-25. The experimental results are shown in Table 5.4. The proposed network based on ResNet-50 achieves

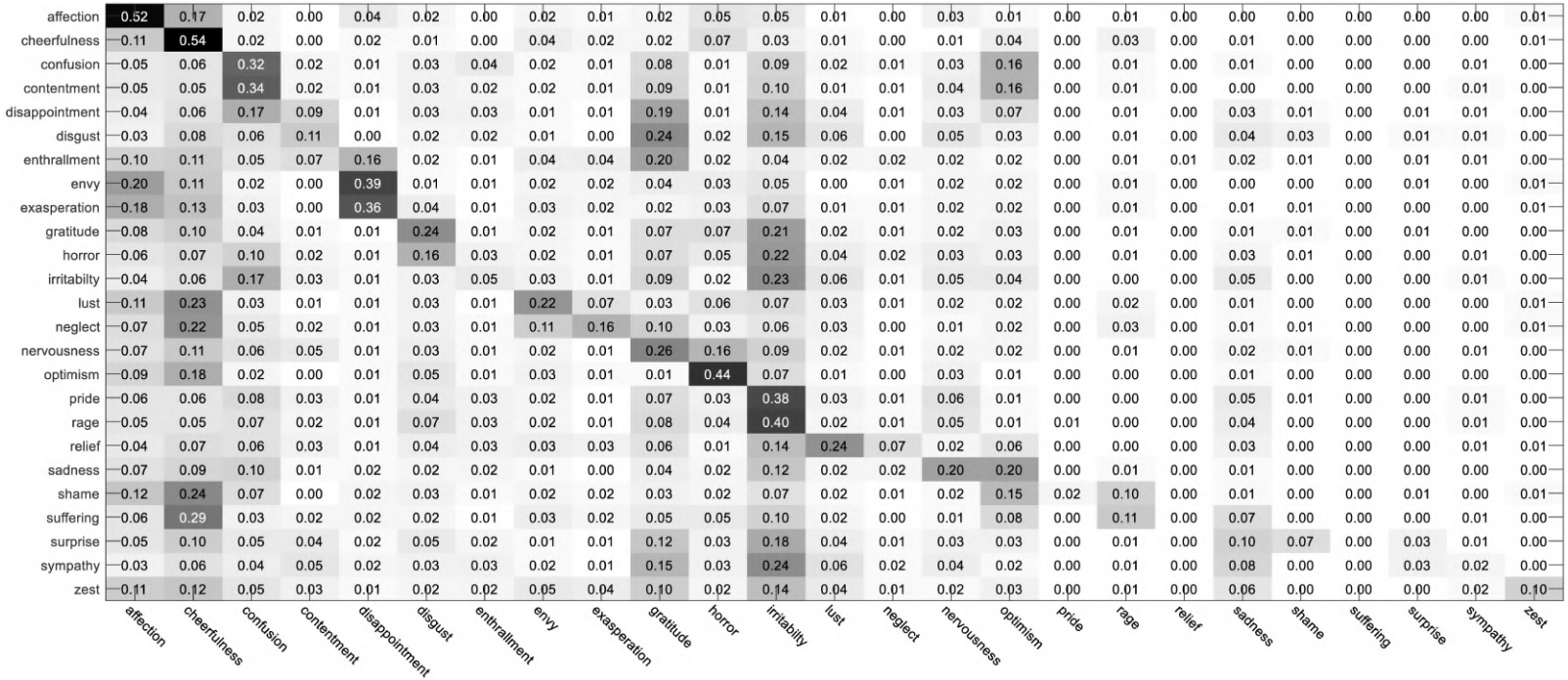


Figure 5.7 : The confusion matrix on WEBEmo-25 using the proposed network based on ResNet-101.

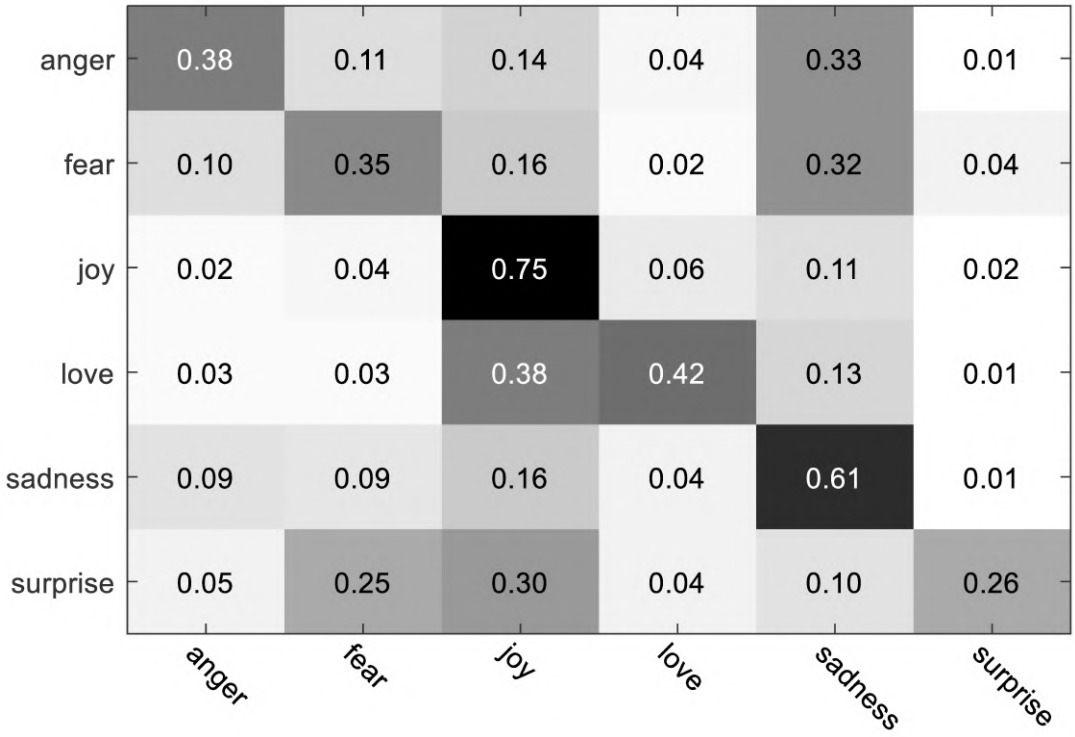


Figure 5.8 : The confusion matrix on WEBEmo-6 using the proposed network based on ResNet-101.

0.0% and 0.74% performance improvements on WEBEmo-6 and WEBEmo-25, respectively, and our network based on ResNet-101 yields 0.0% and 0.87% performance gain on the two datasets, respectively. It is worth noting that our network based on ResNet-50 achieves higher recognition accuracy than the vanilla ResNet-101.

Figure 5.7 shows the the confusion matrix on WEBEmo-25 using the proposed network based on ResNet-101. We observe from Figure 5.7 that most categories are confused with the "irritability" category. The proposed network achieves the highest recognition accuracy of 54.0% for the emotion of cheerfulness. The recognition accuracy is 0 for six categories, *i.e.,* "neglect" "pride", "rage", "relief", "shame" and "suffering". There are totally 19 categories for which the recognition accuracy lower than 5.0%. This shows that these fine-grained emotion categories are difficult to recognize.

Model	Recognition accuracy (%)
Sentibank [7]	44.49
Zhao *et al.* [62]	46.52
DeepSentiBank [65]	53.16
PCNN [66]	56.16
AlexNet [25]	58.3
ResNet-101	65.99
Zhu *et al.* [70]	73.03
Rao *et al.* [8]	75.46
Ours + ResNet-50	**74.84**
Ours ResNet-101	**75.91**

Table 5.3 : Recognition accuracy of the proposed network on FI-8 and comparison with previous work.

Model	Recognition accuracy (%)	
	WEBEmo-6	WEBEmo-25
ResNet-50 (vanilla)	—	31.80
ResNet-101 (vanilla)	—	32.14
Ours + ResNet-50	52.26	32.54
Ours + ResNet-101	**53.88**	**33.01**

Table 5.4 : Recognition accuracy of the proposed network on WEBEmo-6 and WEBEmo-25.

5.3.5 Image sentiment analysis

We further evaluated the proposed network for image sentiment classification. The task of image sentiment classification is to classify an image as positive sentiment or negative sentiment, which means the general attitude or opinion conveyed in the image. We conducted experiments on Ekman-6, FI-8, and WEBEmo. We converted the original

Model	Recognition accuracy (%)		
	Emotion-2	FI-2	WEBEmo-2
Sentibank [64]	—	56.47	—
DeepSentibank [114]	—	64.39	—
PCNN [66]	—	75.34	—
AlexNet [25]	75.88	72.43	—
VGGNet-16 [26]	—	83.05	—
ResNet-50 [71]	—	—	76.65 [71]
ResNet-101	79.78	75.76	—
Zhu *et al.* [70]	—	84.26	—
Curriculum Learning + ResNet-50 [71]	77.72	84.81	81.41 [71]
Yang *et al.* [7]	—	86.35	—
Rao *et al.* [8]	—	87.51	—
Ours + ResNet-50	82.42	90.58	81.94
Ours + ResNet-101	**82.62**	**90.97**	82.47

Table 5.5 : Image sentiment recognition results using the proposed network on Ekman-2, FI-2, and WEBEmo-2, and comparison with previous work.

labels to positive and negative. The converted datasets are denoted as Ekman-2, FI-2, and WEBEmo-2, respectively, in this thesis.

The experimental results and the comparison with previous work are shown in Table 5.5. We see from Table 5.5 that our network based on ResNet-50 achieves 4.72% higher recognition accuracy than vanilla ResNet-50 on Ekman-2. It also outperforms the curriculum learning method by 4.24%. The recognition performance improves barely by changing the backbone network from ResNet-50 to ResNet-101.

5.4 Conclusion

We have proposed an end-to-end network for image emotion recognition by utilizing emotion intensity. To the best of our knowledge, this is the first on semi-supervised emo-

tion intensity prediction for image emotion recognition. The proposed network comprises an emotion intensity prediction stream and a classification stream. The intensity prediction stream, which is built on top of the FPN, inferences emotion intensity maps directly from the input image. The predicted intensity maps are integrated to the classification stream for final emotion recognition. The two streams are trained cooperatively with each other to improve performance. We evaluate the proposed network for image emotion recognition and sentiment analysis on three benchmark datasets. The experimental results show that the proposed network achieves better performance than region-based approaches.

5.A Appendix

The calculation of the gradient of an intensity map is presented in this section. The two filters are defined as derivative kernels to calculate gradients in x direction and in y direction, respectively. In this work, we use the following two Sobel filters:

$$S_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \; S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \tag{5.10}$$

We then run the two Sobel filters over the intensity map $\boldsymbol{f}$, and obtain the following two parts:

$$\begin{aligned} \frac{\partial f}{\partial x} &= \boldsymbol{S}_x \otimes \boldsymbol{f}, \\ \frac{\partial f}{\partial y} &= \boldsymbol{S}_y \otimes \boldsymbol{f}, \end{aligned} \tag{5.11}$$

where $\otimes$ denotes convolution operation.

The gradient of the image is represented as the concatenation of the above two parts, which is given follows:

$$\nabla \boldsymbol{f} = \left[\frac{\partial \boldsymbol{f}}{\partial x}, \frac{\partial \boldsymbol{f}}{\partial y} \right]. \tag{5.12}$$

Chapter 6

Dual Pattern Learning with Adversarial Adaptation

6.1 Introduction

Convolutional neural networks (CNNs) have become a dominant approach for various machine learning applications such as computer vision [25, 27, 116], natural language processing [117–119], and reinforcement learning [120, 121]. Integrating feature learning and classifiers in an end-to-end manner, deep neural networks can learn features automatically from training data without human involvement during training. It has been shown that features learned by CNNs are much discriminative compared to hand-crafted features [25], and that features extracted from a CNN pretrained on a large scale image dataset can be transferred to other visual recognition tasks [77, 100]. Recent progress of deep neural networks in computer vision and machine learning has enabled transformative applications across robotics, healthcare, and security

Researchers have spent much effort in designing network architectures to improve the performance of CNNs. He *et al.* [27] proposed the residual learning framework. This framework eases the training of deep neural networks, and enables them to be considerably deep. Residual networks (ResNets) have led to performance improvement in both visual and non-visual tasks. Huang *et al.* [28] proposed densely connected networks (DenseNets), in which each layer is connected to every other layer in a feed-forward fashion. This architecture substantially reduces the number of parameters, and is highly computationally efficient as a result of feature reuse.

State-of-the-art deep neural networks usually consist of many layers with a large number of parameters. The ResNeXt-19 (8×64d) [29] contains approximately 3×10^7 parameters to model the 5×10^4 images in CIFAR-10 [122]. The VGGNet-16 [26] has around 10^8 parameters to model the 10^6 images in ImageNet [115]. The large number of parameters makes deep networks prone to overfitting; therefore, training deep networks

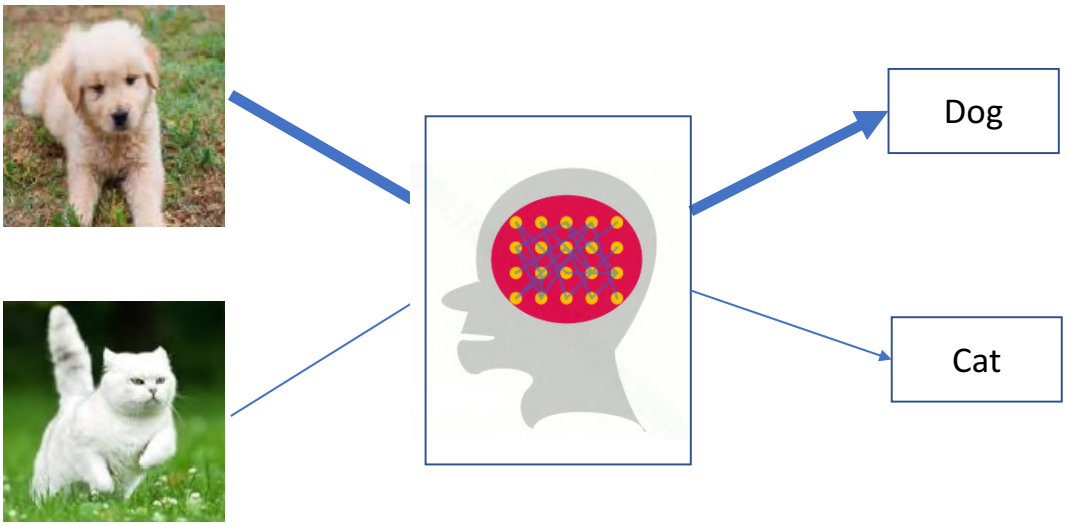


Figure 6.1 : An illustration that shows humans learn knowledge by analyzing dual images. They may have more interest in learning one image than the other image. In this figure, the human is more interested in, or pays more attention to, learning dog (boldness of lines represents interest value).

requires huge amounts of data. However, collecting data and labelling them are laborious work, especially when domain experts are necessary to distinguish between fine-grained visual categories. For some tasks, it is extremely difficult to collect samples.

In this work, we focus on efficient feature learning with limited data using CNNs. We observe that humans can learn knowledge from two given images by analyzing and comparing two images. An illustration is shown in Figure 6.1. Inspired by this observation, we propose a dual pattern learning (DPL) network architecture, referred to as DPLNet in this thesis. Unlike previous deep networks, the DPLNet is trained using image pairs. The number of image pairs can be significantly large even for small datasets. This can help to address the overfitting issue due to lack of training data.

As shown in Figure 6.2, we design two input branches and two loss functions so that the DPLNet can perform dual pattern learning. Two input images are processed by the two input branches in parallel, and feature maps generated by the two branches are then fused together to backbone network. Analyzing dual inputs simultaneously, this

architecture is able to learn robust features. In addition, we observe that people may have more interest in one image than the other image when given two images to learn. This might be due to reasons such as personal preference and/or prior knowledge. Inspired by this observation, we propose to associate the two input branches with two random interest values for learning corresponding images during training. This method can help improve the generalization performance.

Moreover, we introduce to use the adversarial training framework [40] to minimize the domain difference between fused image features and single image features. We refer to the DPLNet with adversarial adaptation as DPLAANet in this thesis. State-of-the-art deep networks can be easily adapted to DPLAANets. We show that our DPLAANets exhibit better performance.

There has been research which uses the mixture of two images as for training deep neural networks, such as Mixup [123] and between class (BC) learning [124]. Unlike their work, we propose a new architecture with dual input branches which are trained using image pairs. Moreover, we incorporate the adversarial training framework to reduce the domain difference between fused image features and single image features.

This chapter provides the following three contributions:

- We propose the DPLAANet architecture towards learning with limited data. The dual input structure of the DPLAANet enables the network to learn robust features by analyzing dual inputs simultaneously. With the dual input structure, we have a large number image pairs to train the network. This helps to address the overfitting issue due to lack of training data.

- The adversarial training framework is incorporated to reduce the domain difference between fused image features and single image features. We show that this method effectively contributes to performance improvement.

- The DPLAANets are evaluated on five benchmark datasets, *i.e.,* CIFAR-10, CIFAR-100, FI-8, Google commands dataset, and MNIST, wherein they lead to performance improvement compared to benchmark networks. The experimental results on subsets of CIFAR-10, CIFAR-10, and MNIST demonstrate that our DPLNets

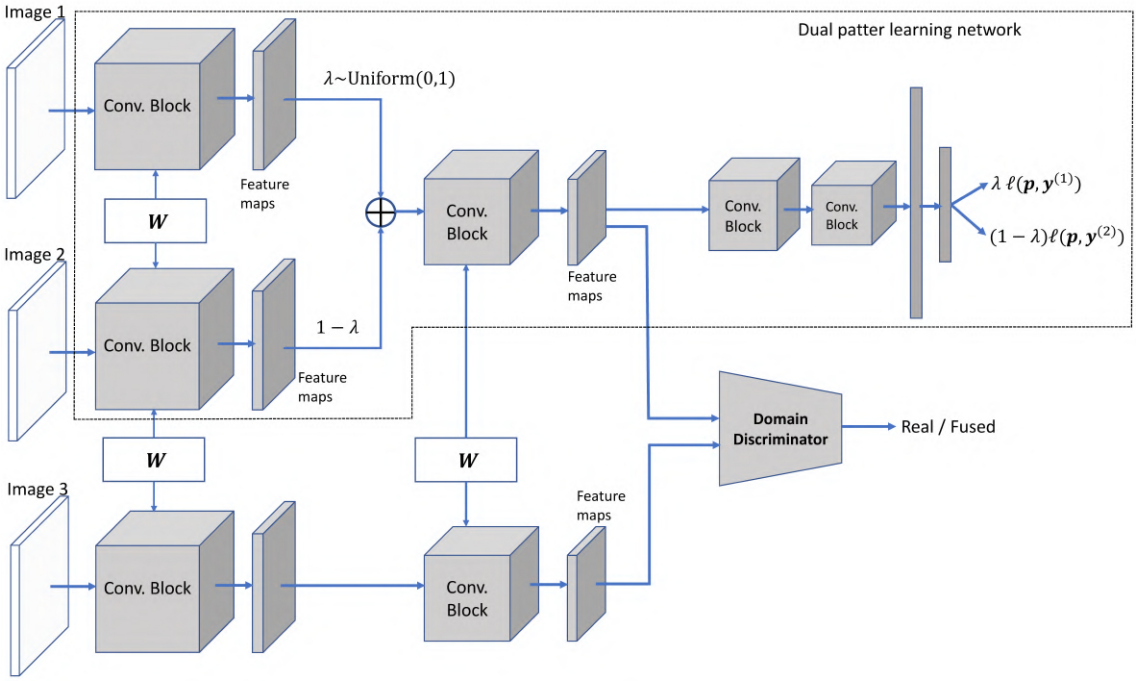


Figure 6.2 : An illustration of the proposed DPLAANet framework. This framework consists of a DPLNet and an adversarial adaptation module. The DPLNet has two input branches which share the same parameters. Feature maps generated by the two input branches are fused together to backbone network. We perform random weighted fusion. A value λ is sampled from the standard uniform distribution as weight for one branch, and $1 - \lambda$ for the other branch. The weight associated with each branch can be considered as an interest value for learning the corresponding image. The adversarial training approach is used to reduce the domain difference between fused image features and real image features.

have outstanding generalization performance on small datasets. Extended experiments on subsets of MNIST show that the architecture with three branches outperforms two branches when training set is extremely small.

6.2 Methodology

6.2.1 Dual pattern learning

Empirical risk minimization (ERM) [125] has been the rule for training neural networks. Given a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with N labelled training data, where x_i and y_i represent the i-th data and its corresponding label, respectively. Under the ERM rule, a neural network f is trained by minimizing the following risk:

$$R(f) = \frac{1}{N} \sum_{i=1}^{N} \ell(f(x_i), y_i), \tag{6.1}$$

where ℓ is a loss function for penalizing errors between prediction result and actual target.

In this thesis, we propose a dual pattern learning architecture. This architecture intends to predict for dual inputs simultaneously. Unlike f for empirical risk minimization, the prediction function g for dual pattern learning takes two samples as input, and is obtained by minimizing dual prediction risk, which is given below:

$$R_{DPL}(f) = \frac{1}{N} \sum\nolimits_{i} \ell(g(x_i^1, x_i^2), y_i^1, y_i^2), \tag{6.2}$$

where the loss function ℓ is to be defined to penalize dual prediction errors. We refer to training neural networks by minimizing Equation (6.2) as empirical dual prediction risk minimization.

An overview of the proposed DPLNet architecture is shown in Figure 6.2. The DPLNet contains several blocks, each of which consists of a number of convolutional layers. Feature maps generated within the same block have the same height and width. We design two input branches and two loss functions so that the DPLNet can perform dual pattern learning. Two input images are processed by the two input branches in parallel, and feature maps generated by the two input branches are then fused together to backbone network which ends with a fully connected layer with softmax. The two input branches share the same parameters. This guarantees that the two input branches generate consistent feature maps for dual inputs, which means that feature maps fused to backbone network are the same regardless of input orders. The DPLNet architecture forces the

network to learn discriminative features by analyzing dual inputs simultaneously; therefore, DPLNets encourage learned features to have large inter-class margins compared to conventional networks.

We use random weighted combination of feature maps generated by the two input branches as input to backbone network during training. In particular, we randomly sample a value λ from the standard uniform distribution, as given below:

$$\lambda \sim Uniform(0, 1). \tag{6.3}$$

In our experiments, we clamp λ into range $[0.2, 0.8]$. This achieves a little bit better performance than without using clamping. The value of λ is used as weight for one branch, and $1 - \lambda$ for the other branch. The two weights can be considered as interest values for learning corresponding input. The fused feature maps is represented as the convex combination of the two sets of feature maps:

$$\mathbf{Conv} = \lambda \mathbf{Conv_1} + (1 - \lambda)\mathbf{Conv_2}, \tag{6.4}$$

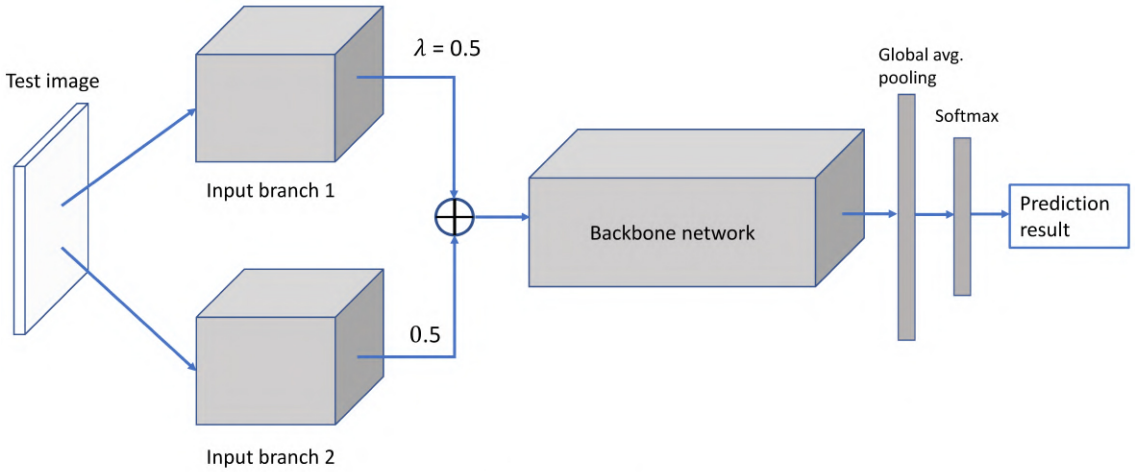
where $\mathbf{Conv_1}$ and $\mathbf{Conv_2}$ represent feature maps generated by the two branches, respectively.

The overall loss function of the DPLNet is defined to penalize dual prediction errors, which is given as follows:

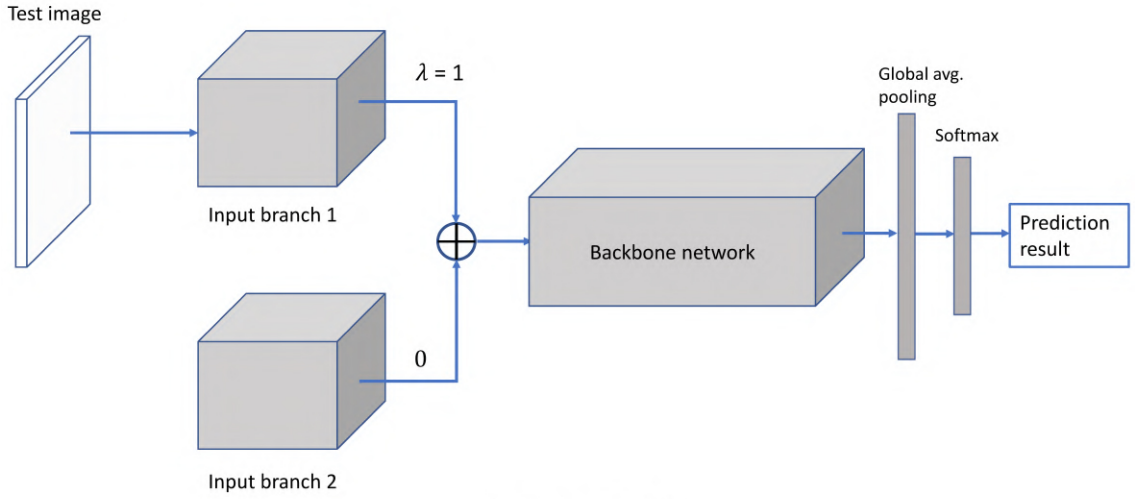
$$L_{DPL} = \lambda \ell_{cls}(\boldsymbol{p}, \boldsymbol{y}^{(1)}) + (1 - \lambda)\ell_{cls}(\boldsymbol{p}, \boldsymbol{y}^{(2)}), \tag{6.5}$$

where λ is the same as in Equation (6.4), $\boldsymbol{p}$ represents the predicted probability, and $\boldsymbol{y}^{(1)}$ and $\boldsymbol{y}^{(2)}$ are one-hot encoding labels for images given as input to the first branch and the second branch, respectively. The cross entropy loss is used as classification loss ℓ_{cls} in this work. If $0.5 < \lambda \leq 0.8$, the DPLNet is more interested in learning the corresponding image than the other image, and it receives more supervision for learning this image. In this case, the other image can be seen as an auxiliary image for learning. If λ is equal to 0.5, the DPLNet has equal interest in learning two input images. If $0.2 \leq \lambda < 0.5$, the DPLNet is more interested in learning the other input images.

At test time, there are two approaches to test an image (see Figure 6.3). The first approach is to pass the test image to the two input branches while setting λ to 0.5. The



(a) Test method 1.



(b) Test method 2.

Figure 6.3 : Two test approaches at test time: (a) Pass a test image to both input branches and set λ to 0.5; (b) Give an image as input to one branch and set corresponding λ to 1 while ignoring the other input branch.

other approach is to pass the image to one input branch and set the corresponding λ to 1 while ignoring the other input branch. The final softmax layer produces a probability distribution over all categories. Because the parameters of the two input branches are tied and feature maps generated by the two branches are fused by convex combination, the fused feature maps are the same for the two test approaches. Therefore, the two test

approaches produces the same prediction result. In our experiments, we use the second approach for testing for time efficiency.

6.2.2 Adversarial domain adaptation

Domain adaptation aims to address the domain shift issue, *i.e.,* training data and testing data have different distributions in feature space. In the dual pattern learning framework, models are trained using random image pairs. We wish to use trained models for single image classification. As introduced in section 6.2.1, our test approach equivalently uses a test image and its duplicate as a pair for testing. We see that the distribution of training data is different from that of testing data. In this work, we propose to add a domain adaptation module to the DPLNet to reduce the domain difference between training data and testing data.

We employ the adversarial learning method for domain adaptation. A discriminator D is defined to distinguish training data distribution from testing data distribution . The adaptation is performed at feature-level instead of at pixel-level [126] (see Figure 6.2). We refer to the DPLNet with adversarial adaptation as DPLAANet in this thesis. We update the discriminator and the DPLNet by alternating two steps. Given a batch of image pairs $\boldsymbol{x} = \{x_i^1, x_i^2\}_{i=1}^B$ and a batch of images $\boldsymbol{t} = \{t_i\}_{i=1}^B$. Let the feature maps for the image pairs and the feature maps for the real images be denoted z_{x_i} and z_{t_i}, respectively. At the first step, the two sets of feature are taken as input to the domain discriminator D, and we update the parameters of the domain discriminator D using the following hinge loss [127]:

$$\begin{aligned} L_D = &\sum\nolimits_i min(0, -1 + D(z_{x_i})) \\ &+ \sum\nolimits_j min(0, -1 - D(z_{t_j})). \end{aligned} \tag{6.6}$$

At the second step, we fix D and update the parameters of the dual pattern learning network using the following loss:

$$L_C = L_{DPL} - \beta \sum\nolimits_i D(z_{x_i}), \tag{6.7}$$

where L_{DPL} is the dual prediction loss (see Equation 6.5) and β is the weight for adjusting

Model	Error rate (%)	
	CIFAR-10	CIFAR-100
ResNet-18	4.96	22.78
ResNet-18 + DPLAANet	**4.45**	**19.95**
ResNet-34	4.86	21.64
ResNet-34 + DPLAANet	**4.10**	**19.28**
ResNet-50	4.62	21.89
ResNet-50 + DPLAANet	**3.96**	**18.75**
ResNet-101	4.44	20.81
ResNet-101 + DPLAANet	**3.81**	**18.32**

Table 6.1 : Test errors (%) on CIFAR-10 and CIFAR-100.

the losses. The spectral normalization method [127] is used to normalize the layers of D. This guarantees the discriminator satisfies the 1-Lipschitz constraint.

6.3 Experiments

We conducted experiments on a diverse of recognition tasks to show that the proposed framework is a general technique to improve the performance of deep networks. We further evaluated DPLAANets on small datasets to show their usefulness when training dataset is small.

6.3.1 CIFAR-10 and CIFAR-100

We first conducted experiments on CIFAR-10 and CIFAR-100 [122]. The CIFAR-10 and CIFAR-100 datasets consist of 32×32 color images categorized into 10 and 100 classes, respectively. The training and testing sets contain 50,000 and 10,000 images, respectively.

We used PyTorch [105] for implementation. We implemented DPLAANets based on four state-of-the-art deep networks, *i.e.,* ResNets, pre-activation ResNets (PreAct ResNets) [128], DenseNets and ResNeXts. The classification accuracies achieved by the vanilla

Model	Error rate (%)	
	CIFAR-10	CIFAR-100
PreAct ResNet-18	4.90	22.48
PreAct ResNet-18 + DPLAANet	**4.02**	**19.44**
PreAct ResNet-34	4.69	21.08
PreAct ResNet-34 + DPLAANet	**3.97**	**18.97**
PreAct ResNet-50	4.52	20.60
PreAct ResNet-50 + DPLAANet	**3.88**	**18.53**
PreAct ResNet-101	4.38	20.51
PreAct ResNet-101 + DPLAANet	**3.74**	**18.02**

Table 6.2 : Test errors (%) on CIFAR-10 and CIFAR-100.

ResNets are used as baselines for comparison. Each input branch of our DPLAANets consists of one block, and the backbone networks consist of two/three blocks. Following [27, 28, 128], we applied zero-padding of four pixels to training images for training DPLAANets based on ResNets, PreAct ResNets, and DenseNets. Following [29], we applied zero-padding of eight pixels to training images for training DPLAANets based on ResNeXts. A 32×32 image was randomly cropped from the padded image or its horizontal flip as input data to train the models. Each channel of input data were normalized to have zero mean and unit variance. We did not use dropout, following the practice in [33]. All models were trained from scratch using SGD for 300 epochs with a mini-batch of 128/64 examples. The learning rate for training DPLNets started from 0.1 and was divided by 10 at epoch 150 and 225. The values of weight decay and momentum were set to 0.0005 and 0.9, respectively. We used the Adam [129] algorithm to train the discriminator. The learning rate for training discriminators was set to 2e-4. At test time, we only evaluated the original 32×32 image. The value of λ was chosen from 0.01, 0.001, 0.0001 and 0.00001.

Model	Error rate (%)	
	CIFAR-10	CIFAR-100
DenseNet-121 ($k = 32$)	4.55	22.0
DenseNet121 + DPLAANet	**4.21**	**18.75**
DenseNet-169 ($k = 32$)	4.46	20.21
DenseNet-169 + DPLAANet	**4.14**	**18.05**

Table 6.3 : Test errors (%) on CIFAR-10 and CIFAR-100. k indicates the growth rate of network.

Model	Error rate (%)	
	CIFAR-10	CIFAR-100
ResNeXt-29 8×64d [29]	3.65	17.77
ResNeXt-29 8×64d + DPLAANet	**3.27**	**16.93**
ResNeXt-29 16×64d [29]	3.58	17.31
ResNeXt-29 16×64d + DPLAANet	**3.08**	**16.66**

Table 6.4 : Test errors (%) on CIFAR-10 and CIFAR-100.

DPLAANets based on ReseNets

We implemented DPLAANets based on four ResNet architectures, *i.e.,* ResNet-18, ResNet-34, ResNet-50, and ResNet-101. The experimental results are shown in Table 6.1. From this table, we observe that as with ResNets, the performance of our DPLAANets improves as the number of layers increases. The DPLAANet based on ResNet-101 achieves the best performance on the two datasets, whereby it achieves 3.81% and 18.32% error rates, respectively. By using the proposed framework, performance improves for the four ResNet architectures, with at least 0.51% and 2.49% performance improvements on the two datasets, respectively. On average, DPLAANets achieve 0.64% and 2.71% performance gains on the two datasets, respectively. The performance improvements yielded by using DPL with adversarial adaptation are higher on CIFAR-10 than on CIFAR-100.

Base model	Method	Error rate (%)	
		CIFAR-10	CIFAR-100
ResNet-18	Vanilla	4.96	22.78
	DPLNet	4.63	20.70
	DPLADNet	4.45	19.95
ResNet-34	Vanilla	4.86	21.64
	DPLNet	4.35	19.83
	DPLAANet	4.10	19.28

Table 6.5 : Ablation study. Performance comparison among DPLAANets, DPLNets, and vanilla ResNets on CIFAR-10 and CIFAR-100.

DPLAANets based on PreAct ReseNets

We investigated DPLAANets based on four PreAct ResNets architectures, *i.e.,* PreAct ResNet-18, PreAct ResNet-34, PreAct ResNet-50, and PreAct ResNet-101. The experimental results are shown in Table 6.2. From this table, we find that using the proposed method helps to improve classification accuracy for the four PreAct ResNet architectures. The DPLAANets based on PreAct ResNet-101 achieve the best performance on the two datasets. The DPLNets exhibit better performance based on PreAct ResNet than based on ResNet.

DPLAANets based on DenseNets

We implemented DPLAANets based on two DenseNet architectures, *i.e.,* DenseNet-121 and DenseNet-169. The experimental results are shown in Table 6.3. We see from Table 6.3 that our DPLAANets achieve better performance than vanilla DenseNets. On average, the DPLAANets achieve 0.33% and 2.71% performance improvements on the two datasets, respectively.

Model	Num. of input branches	Error rate (%)	
		CIFAR-10	CIFAR-100
ResNet-18 + DPLAANet	1 (vanilla)	4.96	22.78
	2	**4.45**	**19.95**
	3	4.71	19.98
ResNet-34 + DPLAANet	1 (vanilla)	4.86	21.64
	2	**4.10**	**19.28**
	3	4.16	19.35

Table 6.6 : Impact of number of input branches on performance. We did not use adversarial adaptation for branch number equal to 1.

DPLAANets based on ResNeXts

We investigated DPLNets based on two ResNeXt architectures, *i.e.,* ResNeXt-29 (8×64d) and ResNeXt-29 (16×64d). The comparison of results between our DPLAANets and vanilla ResNeXts is show in Table 6.4. From this table, we observe that by using dual pattern learning with adversarial adaptation, performance improves compared to original ResNeXts. The DPLAANet based on ResNeXt-29 (16×64d) achieves the best performance on the two datasets, wherein it achieves 3.08% and 16.66% error rates, respectively.

We have seen that the proposed DPLAANet architecture helps to improve performance based on the four types of deep network architectures, *i.e.,* ResNets, PreAct ResNets, DenseNets, and ResNeXts. This demonstrates that the performance of the DPLAANet framework is stable. We observe that the performance improvements achieved by DPLAANets are more significant on CIFAR-100 than on CIFAR-10. In CIFAR-10 and CIFAR-100, each category has 5000 and 500 samples, respectively. The results show that DPLNets are very helpful for small training sets.

We conducted an ablation study to understand how each module of the proposed method contributes performance improvement. We trained DPLNets based on ResNet-

Model	Error rate (%)	
	CIFAR-10	CIFAR-100
ResNet with stochastic depth [130]	5.25	24.98
ResNet-1001 [128]	4.92	22.71
Wide ResNet-28 [131]	4.17	20.50
PyramidNet [132]	4.70	22.77
CliqueNet [133]	5.06	21.83
DCNet-32 [134]	4.75	20.23
ResNet-18 + cutout [135]	3.99	21.96
ResNet-18 + DPLAANet (Ours)	**4.45**	**19.95**
PreAct ResNet-18 + mixup [123]	4.20	21.10
PreAct ResNet-18 + DPL (Ours)	4.16	20.15
PreAct ResNet-18 + DPLAANet (Ours)	**4.02**	**19.44**
ResNeXt-29 + BC+ [124]	**2.81**	17.93
ResNeXt-29 + DPL (Ours)	3.32	16.90
ResNeXt-29 + DPLAA (Ours)	3.08	**16.66**

Table 6.7 : Comparison with previous work on CIFAR-10 and CIFAR-100.

18 and ResNet-34. The performance comparison among ResNets, DPLNets, and D-PLAANets are shown in Table 6.5. We observe that the performance improves 0.42% and 1.95% on the two datasets, respectively, on average by using dual pattern learning. The adversarial adaptation module further yields 0.22% and 0.65% improvements on average on on the two datasets, respectively. The test errors evolutions on CIFAR-10 and CIFAR-100 for ResNets, DPLNets, and DPLAANets are shown in Figure 6.4. We further evaluated the impact of the number of input branches on performance. The experimental results are shown in Table 6.6, from which we see that increasing input branch number from 2 to 3 can not further improve classification accuracy.

Comparison with previous work To show the advantage of the proposed approach, we compared the performance of the proposed approach with recent work on CIFAR-10

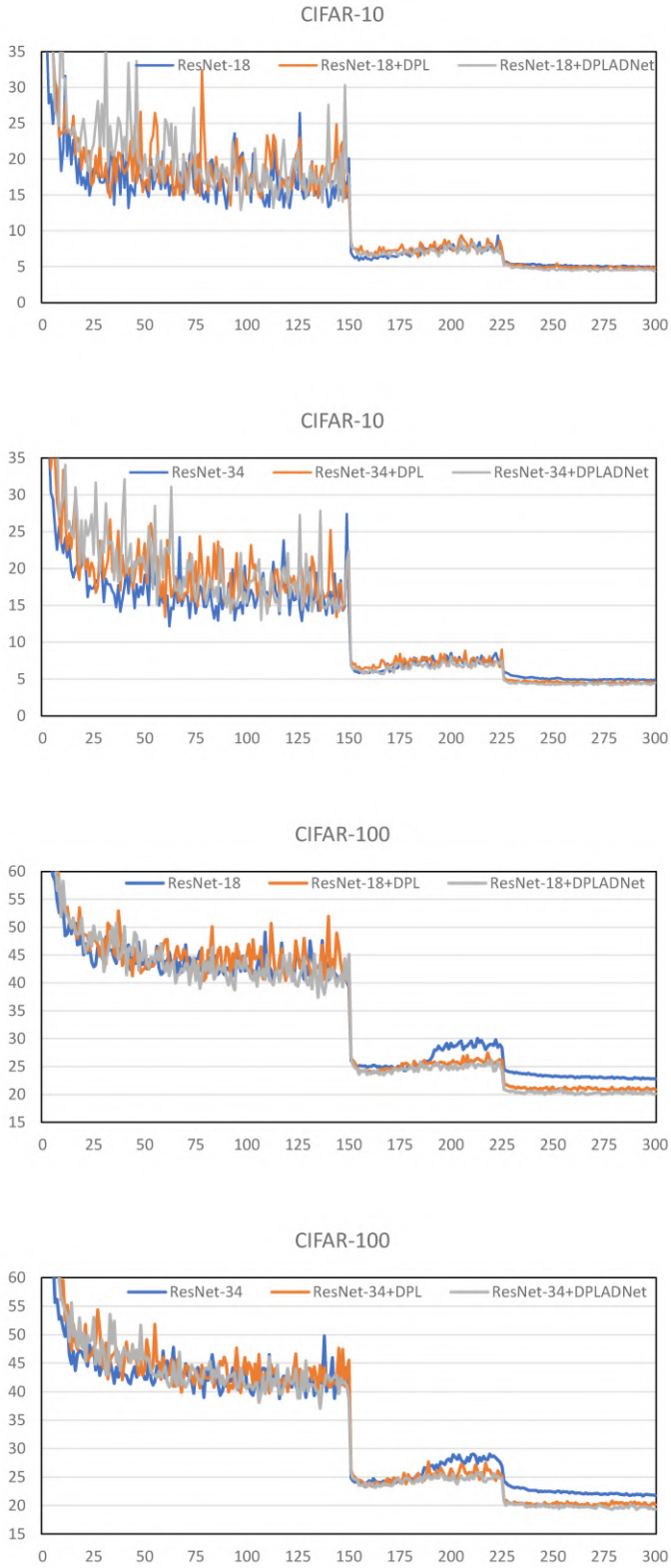


Figure 6.4 : Test errors on CIFAR-10 and CIFAR-100 for ResNets, DPLNets, and D-PLAANets.

and CIFAR-100. The comparison results are shown in Table 6.7.

We see from Table 6.7 that the proposed method achieves comparable performance with the recent work. The ideas behind mixup [123] and between-class (BC) learning [124] are the same. They use a mixture of two images as input to train deep networks; however, the mixture of two images does not visually make sense. Based on PreAct ResNet-18, our DPLNets and DPLAANets achieve low error rates on the two datasets compared to the mixup method. While the proposed method does not perform as good as the BC method on CIFAR-10 based on ResNeXt-29, our DPLNets and DPLAANets achive 1.03% and 1.27% lower error rates on CIFAR-100 than the BC+ method.

6.3.2 Image emotion recognition

For image emotion recognition, experiments were carried out on the FI-8 dataset [59]. This dataset was collected from Flickr and Instagram. There are totally 23,308 images labelled with eight emotion categories. The FI-8 dataset is randomly split into 80% training, 5% validation, and 15% testing sets. In our experiments, all training images were resized with the size of the shorter side equal to 256 while maintaining the original aspect ratio. A 224×224 image was randomly cropped from original image or its horizontal flip as input data to networks. Each channel of input data was normalized to have zero mean and unit variance. At test time, the network made a prediction by cropping 10 regions of the size of 224×224 (four corners and one center, and their horizontal flip) from a test image, and averaging the predictions made by the network's softmax layer on the ten patches.

We implemented DPLAANets based on ResNets which are pre-trained on ImageNet. Each input branch of the DPLAANets consists of one block, and the backbone networks consist of three blocks. We trained the DPLAANets using SGD for 90 epochs with a mini-batch of size 128/64. The values of weight decay and momentum were set to 0.0001 and 0.9, respectively. The learning rate for training DPLNets started from 0.1 and was divided by 10 after 30 and 60 epochs. We used the Adam algorithm to train the discriminator. The learning rate was set to 2e-4. The value of λ was chosen from 0.01, 0.001, 0.0001 and 0.00001. The experimental results are shown in Table 6.8. From this table, we

Model	Recognition accuracy (%)
ResNet-18	64.48
ResNet-18 + DPLAANet	**66.56**
ResNet-34	65.08
ResNet-34 + DPLAANet	**68.17**
ResNet-50	65.99
ResNet-50 + DPLAANet	**68.60**
ResNet-101	66.56
ResNet-101 + DPLAANet	**69.24**

Table 6.8 : Recognition accuracies (%) on FI-8.

Model	Validation set	Test set
LeNet-5	9.8	10.3
LeNet-5 + Mixup ($\alpha = 0.1$)	10.1	10.8
LeNet-5 + DPLAANet	**8.2**	**8.5**

Table 6.9 : Error rates (%) on the Google commands dataset.

Model	Test set
LeNet-5 [no distortions] [136]	0.95
LeNet-5 [huge distortions] [136]	0.85
LeNet-5 [distortions] [136]	0.8
LeNet-5 [no distortions] + DPLAANet	**0.52**

Table 6.10 : Error rates (%) on MNIST.

observe that as with ResNets, the performance of DPLAANets improves as the number of layers increases. The DPLAANets achieve better performance than original ResNets. The DPLAANet based on ResNet-101 achieves the best classification accuracy of 69.24%

Num. of samples per category	100	200	300	400	500
ResNet-18	45.76	34.70	27.57	21.07	18.46
ResNet-18 + DPLAANet	**38.02**	**25.14**	**18.84**	**16.25**	**14.47**
PreAct ResNet-18	45.32	33.61	25.62	20.71	18.38
PreAct ResNet-18 + DPLAANet	**33.45**	**22.97**	**17.89**	**15.60**	**13.91**

Table 6.11 : Error rates (%) on subsets of CIFAR-10.

Num. of samples per category	100	200	300	400	500 (Full dataset)
ResNet-18	44.65	32.58	27.06	24.22	22.78
ResNet-18 + DPLAANet	**38.21**	**27.79**	**23.02**	**20.80**	**19.95**
PreAct ResNet-18	55.95	33.03	28.16	25.05	22.48
PreAct ResNet-18 + DPLAANet	**37.63**	**26.96**	**23.34**	**22.92**	**19.44**

Table 6.12 : Error rates (%) on subsets of CIFAR-100.

Num. of samples per category	10	20	30	50	100
LeNet-5	28.73	16.97	12.81	8.31	4.58
LeNet-5 + DPLAANet	20.57	13.89	10.21	6.73	3.94
LeNet-5 + TPLAANet	20.04	13.83	9.98	6.62	3.74

Table 6.13 : Error rates (%) on subsets of MNIST. TPLAANet represents triple pattern learning with adversarial adaptation networks, in which three input branches are used.

on this dataset. Using the proposed DPLAANet framework yields an average of 2.63% performance improvement.

6.3.3 Google commands dataset

We further conducted experiments on speech data. We used the Google commands dataset [137]. This dataset consists of 65,000 utterances which were recorded by thou-

sands of different people. There are totally 30 categories. Each utterance is about one-second long and belongs to one out of 30 short words, such as “yes”, “no”, “down”, and “left”. Following the work of [123], we down-sampled from the original waveforms with the sampling rate equal to 16 kHz, and extracted normalized spectrograms. We applied zero-padding to the spectrograms such that their size equal to 160×101. We implemented DPLAANet based on LeNet-5 [136]. Each input branch consists of a convolutional and a subsampling layer, and feature maps generated by two input branches are then fused to backbone network. The first fully connected layer contains 16280 neurons, and the second fully connected layer contains 1000 neurons. The models were trained using SGD with a mini-batch of 100 examples. The learning rated started at 0.001 and was divided by 10 after 50 epochs. The discriminator was trained using the Adam [10] algorithm with learning rate set to 2e-4. The experimental results are shown in Table 6.9. From this table, we see that our DPLAANet yields 1.6% and 1.8% performance improvement on the validation set and the testing set, respectively.

6.3.4 MNIST classification

The MNIST digit dataset consists of 60,000 training and 10,000 testing images of ten handwritten digits (“0” to “9”), each with 28×28 pixels. We implemented DPLAANet based on LeNet-5 [136]. The LeNet-5 consists of two convolutional layers, which are followed by subsampling layers, and three fully connected layers with a final softmax. In our implementation, each input branch contains a convolutional and a subsampling layer, and the backbone network consists of a convolutional layer, a subsampling layer, and three fully connected layers. The model was trained from scratch using Adam for 500 epochs with a mini-batch of 128 examples. The learning rate was set to 0.001 and 2e-4 for training the DPLNet and the discriminator, respectively. The experimental results are shown in Table 6.10. From this table, we find the DPLAANet achieves 0.43% higher performance than original LeNet-5. Our approach also achieve better performance than LeNet-5 trained with distortions of input.

6.3.5 Experiments on Small Datasets

We conducted experiments on small datasets to demonstrate the advantage of D-PLAANets when training data are extremely small. We used subsets of CIFAR-10, CIFAR-100, and MNIST. For experiments on subsets of CIFAR-10 and CIFAR-100, we randomly selected 100, 200, 300, 400, and 500 training samples from each category. The DPLAANets were implemented based on ResNet-18 and PreAct ResNet-18. The parameter setting and the training procedures were the same as in section 6.3.1. For experiments on subsets of MNIST, we randomly selected 10, 20, 30, 50, and 100 training samples from each category. We used the same DPLAANet structure, parameter setting, and training procedures as in section 6.3.4, excepted that we used a smaller batch size of 50. The experimental results, which are calculated by averaging three runs, are shown in Table 6.11, Table 6.12, and Table 6.13, respectively.

From Table 6.11, we find that the DPLAANet based on ResNet-18 and the DPLAANet based on PreAct ResNet-18 yield the heighest performance improvements of 9.56% and 11.87%, respectively, with each category has 200 and 100 training samples, respectively, on CIFAR-10. As the number of training samples in each category increases from 200 to 500, the performance improvement decreases. Overall, they achieve at least 3.99% and 4.47% performance improvements, respectively. The DPLAANets achieve an average of 4.30% and 6.87% performance improvement on subsets of CIFAR-100, respectively, compared to original networks. The performance improvements are higher on CIFAR-10 than on CIFAR-100. This is because CIFAR-100 has more categories, which makes network difficult to distinguish from each other. The DPLAANet yields an average of 3.76% performance improvement on MNIST (see Table 6.13). Moreover, we see from Table 6.13 that increasing the number of input branches from 2 to 3 further improves performance.

A general observation from the three tables is that the performance improvement yielded by DPLAANets is high on subsets with each category has a small number of samples. The experimental results show that the proposed dual patter learning with adversarial adaptation framework is very much helpful when training data are limited. This framework would be promising for other tasks in which training samples are extremely

difficult to collect.

6.4 Conclusion

In this chapter, we have presented the DPLAANet architecture which comprises a DPLNet and an adversarial adaptation module. This DPLNet can learn robust features by analyzing dual inputs simultaneously compared to conventional networks. The dual input structure of the DPLNet enables the network to have a large number of image pairs to train the network, which can help address the overfitting issue due to limited training data. The adversarial training approach is incorporate to reduce the domain difference between fused image features and single image features. We evaluated DPLAANets on on a diverse of classification tasks including image classification, image emotion recognition, handwritten digit recognition and speech recognition. The experimental results show that DPLAANets could lead to performance improvement over state-of-the-art networks. The experimental results on small datasets show that our DPLAANets have good generalization performance when limited training samples are available. This chapter provides a promising approach for applying deep neural networks to tasks in which training samples are extremely difficult to collect.

Chapter 7

Conclusion and Future Work

In this chapter, we present a summary of our work in this thesis and discuss potential future work.

7.1 Conclusions

The majority of this thesis has been focused on emotion recognition in videos and images. Regarding emotion recognition in videos, we conducted two studies: 1) We investigated kernelized features for emotion recognition in videos. Our kernelized features are transformed from spatial features by a kernel function. Compared to spatial image features, kernelized features have shown to be robust. 2) We dealt with the challenge that different video frames could invoke different levels of emotion. We used a deep network for frame-level emotion intensity prediction and employ the predicted frame-level emotion intensities to generate video-level features.

As for emotion recognition in images, we researched on utilizing emotion intensity to improve recognition performance. We proposed an end-to-end network architecture which is trained with pseudo intensity maps. The proposed network has shown to be superior compared to existing approaches including region-based methods. In our fourth work, we addressed the problem that deep networks need to be trained on a large-scale dataset. We introduced a network architecture which is trained with image pairs. The domain adaption method is incorporated to the network to reduce domain difference between training data and test data.

We summarize our main contributions as follows:

- We proposed a discriminative video representation approach using the kernel method for emotion recognition. To generate kernelized features, we constructed a polyno-

mial kernel function based on rewritten the discrete Fourier transformation as a linear kernel. Moreover, this is the first work to apply the denosing method for suppressing the impact of noise contained in videos. Our approach was evaluated on two challenging benchmark datasets. The experimental results show that our kernelized features exhibit superior performance compared to spatial features.

- We proposed a weighted sum pooling method to generate video-level features taking frame-level emotion intensity into account. To the best of our knowledge, this is the first work using frame-level emotion intensity for video-level feature generation. The experimental results demonstrated the effectiveness of the proposed approach.

- We proposed an end-to-end deep neural network for emotion recognition in images which is composed of an intensity prediction stream and a classification stream. The predicted intensity maps are integrated to the classification stream for emotion recognition. The proposed network has demonstrated to be effective to improve recognition performance for both image emotion recognition and sentiment analysis.

- We introduced the DPLAANet architecture which comprises a DPLNet and an adaptation module. The proposed network can help to address the overfitting issue due to limited training data. The experimental results on a diverse of classification tasks showed that our network is general to improve recognition performance. The experimental results on small datasets demonstrated that the proposed network is very effective when limited training data are available.

7.2 Future Work

While much progress in both video representation and image recognition has been made, there remain a number of directions in our future research, some of which we'll discuss below.

- We have shown that kernelized features are effective to improve emotion recognition performance in videos. In our future work, we will evaluate kernelized features

for other video analysis tasks such as event detection, human action detection, and video captioning.

- In this thesis, we used off-the-shelf deep networks for feature extraction. Using transferred features might restrict recognition performance. In our future work, we could investigate discovering representative frames and fine-tuning deep networks on selected frames, and evaluate recognition performance using the fine-tuned networks for feature extraction. Another direction would be collecting a large-scale video emotion dataset and developing end-to-end deep networks for emotion recognition.

- It has been demonstrated that utilizing emotion intensity helps to improve image emotion recognition performance. In this thesis, our network for emotion intensity prediction is built on the FPN. Recent studies have shown that NAS approaches can produce high performance network architectures for both image classification and object detection. Developing new NAS methods to search for a robust network for emotion intensity prediction would be a promising research direction.

Bibliography

[1] P. J. Lang, "A bio-informational theory of emotional imagery," *Psychophysiology*, vol. 16, no. 6, pp. 495–512, 1979.

[2] P. J. Lang, M. M. Bradley, and B. N. Cuthbert, "Emotion, motivation, and anxiety: Brain mechanisms and psychophysiology," *Biological psychiatry*, vol. 44, no. 12, pp. 1248–1263, 1998.

[3] H. Yang, U. Ciftci, and L. Yin, "Facial expression recognition by de-expression residue learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2168–2177.

[4] F. Zhang, T. Zhang, Q. Mao, and C. Xu, "Joint pose and expression modeling for facial expression recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3359–3368.

[5] L. Pang, S. Zhu, and C.-W. Ngo, "Deep multimodal learning for affective analysis and retrieval," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 2008–2020, 2015.

[6] Q. You, J. Luo, H. Jin, and J. Yang, "Cross-modality consistent regression for joint visual-textual sentiment analysis of social multimedia," in *Proceedings of the Ninth ACM international conference on Web search and data mining*. ACM, 2016, pp. 13–22.

[7] J. Yang, D. She, M. Sun, M.-M. Cheng, P. L. Rosin, and L. Wang, "Visual sentiment prediction based on automatic discovery of affective regions," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2513–2525, 2018.

[8] T. Rao, X. Li, H. Zhang, and M. Xu, "Multi-level region-based convolutional neu-

ral network for image emotion classification," *Neurocomputing*, vol. 333, pp. 429–439, 2019.

[9] A. Hanjalic, "Extracting moods from pictures and sounds: Towards truly personalized tv," *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 90–100, 2006.

[10] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.

[12] P. Ekman, "Emotion in the human face (2e éd.)," 1982.

[13] R. Plutchik and H. Kellerman, *Emotion: theory, research and experience*. Academic press New York, 1986, vol. 3.

[14] W. G. Parrott, *Emotions in social psychology: Essential readings*. Psychology Press, 2001.

[15] J. Posner, J. A. Russell, and B. S. Peterson, "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology," *Development and psychopathology*, vol. 17, no. 3, pp. 715–734, 2005.

[16] H. Schlosberg, "Three dimensions of emotion." *Psychological review*, vol. 61, no. 2, p. 81, 1954.

[17] L. F. Barrett, "Discrete emotions or dimensions? the role of valence focus and arousal focus," *Cognition & Emotion*, vol. 12, no. 4, pp. 579–599, 1998.

[18] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 english lemmas," *Behavior research methods*, vol. 45, no. 4, pp. 1191–1207, 2013.

[19] S. Benini, L. Canini, and R. Leonardi, "A connotative space for supporting movie affective recommendation," *IEEE Transactions on Multimedia*, vol. 13, no. 6, pp. 1356–1370, 2011.

[20] M. Solli and R. Lenz, "Color based bags-of-emotions," in *Computer Analysis of Images and Patterns*. Springer, 2009, pp. 573–580.

[21] K. Sun, J. Yu, Y. Huang, and X. Hu, "An improved valence-arousal emotion space for video affective content representation and recognition," in *2009 IEEE International Conference on Multimedia and Expo*. IEEE, 2009, pp. 566–569.

[22] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[23] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[28] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer*

vision and pattern recognition, vol. 1, no. 2, 2017, p. 3.

[29] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5987–5995.

[30] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by gan improve the person re-identification baseline in vitro," *arXiv preprint arXiv:1701.07717*, vol. 3, 2017.

[31] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[33] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[34] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[35] Y. Xu, S. J. Pan, H. Xiong, Q. Wu, R. Luo, H. Min, and H. Song, "A unified framework for metric transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 6, pp. 1158–1171, 2017.

[36] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 193–200.

[37] J. Hoffman, E. Tzeng, J. Donahue, Y. Jia, K. Saenko, and T. Darrell, "One-shot adaptation of supervised deep convolutional models," *arXiv preprint arXiv:1312.6204*, 2013.

[38] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," *arXiv preprint arXiv:1711.03213*, 2017.

[39] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[41] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3801–3809.

[42] Y. Zhang, H. Tang, K. Jia, and M. Tan, "Domain-symmetric networks for adversarial domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5031–5040.

[43] H.-B. Kang, "Affective content detection using hmms," in *Proceedings of the eleventh ACM international conference on Multimedia*. ACM, 2003, pp. 259–262.

[44] Z. Rasheed, Y. Sheikh, and M. Shah, "On the use of computable features for film classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 1, pp. 52–64, 2005.

[45] H. L. Wang and L.-F. Cheong, "Affective understanding in film," *IEEE Transactions on circuits and systems for video technology*, vol. 16, no. 6, pp. 689–704, 2006.

[46] Y. Baveye, J.-N. Bettinelli, E. Dellandréa, L. Chen, and C. Chamaret, "A large video database for computational models of induced emotion," in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*. IEEE, 2013, pp. 13–18.

[47] M. Xu, C. Xu, X. He, J. S. Jin, S. Luo, and Y. Rui, "Hierarchical affective content analysis in arousal and valence dimensions," *Signal Processing*, vol. 93, no. 8, pp. 2140–2150, 2013.

[48] Y.-G. Jiang, B. Xu, and X. Xue, "Predicting emotions in user-generated videos." in *AAAI*, 2014, pp. 73–79.

[49] L.-J. Li, H. Su, L. Fei-Fei, and E. P. Xing, "Object bank: A high-level image representation for scene classification & semantic feature sparsification," in *Advances in neural information processing systems*, 2010, pp. 1378–1386.

[50] L. Torresani, M. Szummer, and A. Fitzgibbon, "Efficient object category recognition using classemes," *Computer Vision–ECCV 2010*, pp. 776–789, 2010.

[51] C. Chen, Z. Wu, and Y.-G. Jiang, "Emotion in context: Deep semantic feature fusion for video emotion recognition," in *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 2016, pp. 127–131.

[52] B. Xu, Y. Fu, Y.-G. Jiang, B. Li, and L. Sigal, "Video emotion recognition with transferred deep feature encodings," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. ACM, 2016, pp. 15–22.

[53] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[54] B. Xu, Y. Fu, Y.-G. Jiang, B. Li, and L. Sigal, "Heterogeneous knowledge transfer in video emotion recognition, attribution and summarization," *IEEE Transactions on Affective Computing*, 2016.

[55] C. Li, J. Wang, H. Wang, M. Zhao, W. Li, and X. Deng, "Visual-texual emotion analysis with deep coupled video and danmu neural networks," *arXiv preprint arXiv:1811.07485*, 2018.

[56] B. Xu, Y. Zheng, H. Ye, C. Wu, H. Wang, and G. Sun, "Video emotion recognition with concept selection," in *2019 IEEE International Conference on Multimedia*

and Expo (ICME). IEEE, 2019, pp. 406–411.

[57] P. J. Lang, "International affective picture system (iaps): Affective ratings of pictures and instruction manual," *Technical report*, 2005.

[58] J. Machajdik and A. Hanbury, "Affective image classification using features inspired by psychology and art theory," in *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 2010, pp. 83–92.

[59] Q. You, J. Luo, H. Jin, and J. Yang, "Building a large scale dataset for image emotion recognition: The fine print and the benchmark." in *AAAI*, 2016, pp. 308–314.

[60] X. Lu, P. Suryanarayan, R. B. Adams Jr, J. Li, M. G. Newman, and J. Z. Wang, "On shape and the computability of emotions," in *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 2012, pp. 229–238.

[61] T. Rao, M. Xu, H. Liu, J. Wang, and I. Burnett, "Multi-scale blocks based image emotion classification using multiple instance learning," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 634–638.

[62] S. Zhao, Y. Gao, X. Jiang, H. Yao, T.-S. Chua, and X. Sun, "Exploring principles-of-art features for image emotion recognition," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 47–56.

[63] J. Yuan, S. Mcdonough, Q. You, and J. Luo, "Sentribute: image sentiment analysis from a mid-level perspective," in *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*. ACM, 2013, p. 10.

[64] D. Borth, T. Chen, R. Ji, and S.-F. Chang, "Sentibank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content," in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 459–460.

[65] T. Chen, D. Borth, T. Darrell, and S.-F. Chang, "Deepsentibank: Visual sentiment concept classification with deep convolutional neural networks," *arXiv preprint*

arXiv:1410.8586, 2014.

[66] Q. You, J. Luo, H. Jin, and J. Yang, "Robust image sentiment analysis using progressively trained and domain transferred deep networks," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[67] T. Rao, X. Li, and M. Xu, "Learning multi-level deep representations for image emotion classification," *Neural Processing Letters*, pp. 1–19, 2016.

[68] R. Salakhutdinov and G. E. Hinton, "Deep boltzmann machines." in *AISTATS*, vol. 1, 2009, p. 3.

[69] J. Yang, D. She, and M. Sun, "Joint image emotion classification and distribution learning via deep convolutional neural network." in *IJCAI*, 2017, pp. 3266–3272.

[70] X. Zhu, L. Li, W. Zhang, T. Rao, M. Xu, Q. Huang, and D. Xu, "Dependency exploitation: a unified cnn-rnn approach for visual emotion recognition," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 3595–3601.

[71] R. Panda, J. Zhang, H. Li, J.-Y. Lee, X. Lu, and A. K. Roy-Chowdhury, "Contemplating visual emotions: Understanding and overcoming dataset bias," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[72] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1933–1941.

[73] Y. Li, W. Li, V. Mahadevan, and N. Vasconcelos, "Vlad3: Encoding dynamics of deep features for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1951–1960.

[74] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3034–3042.

[75] L. Zhu, Z. Xu, and Y. Yang, "Bidirectional multirate reconstruction for temporal modeling in videos," *arXiv preprint arXiv:1611.09053*, 2016.

[76] X. Chang, Z. Ma, Y. Yang, Z. Zeng, and A. G. Hauptmann, "Bi-level semantic representation analysis for multimedia event detection," *IEEE transactions on cybernetics*, vol. 47, no. 5, pp. 1180–1197, 2017.

[77] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition." in *Icml*, vol. 32, 2014, pp. 647–655.

[78] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[79] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *arXiv preprint arXiv:1703.06870*, 2017.

[80] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2064–2072.

[81] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[82] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[83] B. Schölkopf and A. J. Smola, "Learning with kernels: support vector machines, regularization, optimization, and beyond (adaptive computation and machine learning)," 2001.

[84] A. Krause and C. E. Guestrin, "Near-optimal nonmyopic value of information in graphical models," *arXiv preprint arXiv:1207.1394*, 2012.

[85] D. Alpay, *Reproducing kernel spaces and applications*. Birkhäuser, 2012, vol. 143.

[86] B. Schölkopf and A. J. Smola, "Learning with kernels. 2002," 2002.

[87] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution," *Communications on pure and applied mathematics*, vol. 59, no. 6, pp. 797–829, 2006.

[88] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3360–3367.

[89] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[90] G. B. Folland, *Fourier analysis and its applications*. American Mathematical Soc., 1992, vol. 4.

[91] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE transactions on acoustics, speech, and signal processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[92] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.

[93] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative cnn video representation for event detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1798–1807.

[94] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[95] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition*

(CVPR), 2010 IEEE Conference on. IEEE, 2010, pp. 3304–3311.

[96] M. Aharon, *Overcomplete dictionaries for sparse representation of signals*. Technion-Israel Institute of Technology, Faculty of Computer Science, 2006.

[97] G. Li, Y. Xie, T. Wei, K. Wang, and L. Lin, "Flow guided recurrent neural encoder for video salient object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3243–3252.

[98] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[99] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[100] H. Zhang and M. Xu, "Recognition of emotions in user-generated videos with kernelized features," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2824–2835, 2018.

[101] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, vol. 1, no. 2, 2017, p. 4.

[102] A. Saxena, M. Sun, and A. Y. Ng, "Learning 3-d scene structure from a single still image," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[103] I. Sobel and G. Feldman, "A 3x3 isotropic gradient operator for image processing," *Pattern Classification and Scene Analysis*, 1968.

[104] J. Hu, M. Ozay, Y. Zhang, and T. Okatani, "Revisiting single image depth estimation: toward higher resolution maps with accurate object boundaries," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1043–1051.

[105] A. Paszke, S. Gross, S. Chintala, and G. Chanan, "Pytorch," 2017.

[106] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[107] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in neural information processing systems*, 2014, pp. 487–495.

[108] H. Liu, M. Xu, J. Wang, T. Rao, and I. Burnett, "Improving visual saliency computing with emotion intensity," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 6, pp. 1201–1213, 2016.

[109] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[110] X. Zhang, Y. Wei, J. Feng, Y. Yang, and T. S. Huang, "Adversarial complementary learning for weakly supervised object localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1325–1334.

[111] K. Li, Z. Wu, K.-C. Peng, J. Ernst, and Y. Fu, "Tell me where to look: Guided attention inference network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9215–9223.

[112] B.-U. Lee, H.-G. Jeon, S. Im, and I. S. Kweon, "Depth completion with deep geometry and context guidance."

[113] P. Edman, W. V. Friesen, and P. Ellsworth, "Emotion in the human face: Guidelines for research and an integration of findings," 1972.

[114] J. A. Mikels, B. L. Fredrickson, G. R. Larkin, C. M. Lindberg, S. J. Maglio, and P. A. Reuter-Lorenz, "Emotional category data on images from the international affective picture system," *Behavior research methods*, vol. 37, no. 4, pp. 626–630, 2005.

[115] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[116] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, 2017, pp. 3859–3869.

[117] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[118] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[119] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models." in *AAAI*, 2016, pp. 2741–2749.

[120] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3357–3364.

[121] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning." in *AAAI*, 2017, pp. 2140–2146.

[122] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[123] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[124] Y. Tokozume, Y. Ushiku, and T. Harada, "Between-class learning for image classification," *arXiv preprint arXiv:1711.10284*, 2017.

[125] V. Vapnik, *Statistical learning theory. 1998*. Wiley, New York, 1998.

[126] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2107–2116.

[127] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[128] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 630–645.

[129] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[130] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European conference on computer vision*. Springer, 2016, pp. 646–661.

[131] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[132] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 6307–6315.

[133] Y. Yang, Z. Zhong, T. Shen, and Z. Lin, "Convolutional neural networks with alternately updated clique," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2413–2422.

[134] A. DeFonzo and J. Tauc, "Decoupled networks: An approach to low symmetry vibrational problems," *Solid State Communications*, vol. 18, no. 8, pp. 937–940, 1976.

[135] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[136] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[137] P. Warden, "Launching the speech commands dataset," https://research.googleblog.com/2017/08/launching-speech-commands-dataset.html.