Feature Learning Boosts Network Performance

by

Shiqi Wang
Lakehead University

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTERS

in the Department of Computer Science

Lakehead University

Feature Learning Boosts Network Performance

by

Shiqi Wang
Lakehead University

ABSTRACT

Features are an important part of machine learning. Features are often the reduced-dimensional representation of input data, feature calculation, extraction, and fusion directly affect the final result of the network. This thesis proposes a total of 3 feature learning methods, and all of them have achieved good results.

In the chapter four of this thesis, the method of video feature extraction is mainly introduced. In this thesis, 3D pre-trained CNN model is used to extract features from continuous video frames, and the feature data is input into the classifier with subnetwork node, and good results are obtained. And the experimental results show that the pre-trained model effectively reduces the training time and improves the testing accuracy. 3D CNN can fully extract the temporal and spatial features of objects. For the subsequent classification accuracy has played a good effect.

Chapter three and five are mainly focus on EEG-based signal emotion recognition. EEG data is a one-dimensional signal, but because the human cerebral cortex has different feedback areas for different emotions, the data of EEG data on different electrode channels are spatially related. And different channels of data have a positive impact on different emotional states. Therefore, this chapter improves the accuracy of the test results by converting the one-dimensional data of the EEG into a two-dimensional matrix and adding weights to the corresponding electrodes to extract spatial features and fuse them with frequency-domain features.

Since the EEG signal is continuous in time, the time-domain features also have a great effect on the final sentiment classification result. Therefore, the fifth chapter adds the extraction of time-domain features based on the fourth chapter. The final accuracy of emotion recognition has been significantly improved.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **vii**

List of Figures **viii**

Acknowledgements **x**

Acknowledgements **xi**

1 Introduction **1**

1.1 Overview 1

1.2 Motivation 3

1.3 Problem Description 4

1.4 Contribution 5

2 Literature Review **6**

2.1 Overview 6

2.2 Hierarchical Network with Subnetwork Nodes 7

2.2.1 Feature Extractor 7

2.2.2 Classifier 8

2.3 Transfer Learning 9

3 EEG-Based Emotion Recognition Using CNN Auto-Encoder and Hierarchical Network with Subnetwork Node **12**

3.1 Overview 13

3.2 Introduction 13
3.3 Method 15
3.3.1 Differential Entropy Features 15
3.3.2 Making Images from EEG Features 16
3.3.3 CNN Auto-Encoder with ResBlock 18
3.3.4 Hierarchical Network with Subnetwork Nodes 20
3.3.5 Feature Fusion 20
3.4 Experiments 21
3.4.1 Dataset 21
3.4.2 Environment Setting 23
3.4.3 Weight Coefficient Test 24
3.4.4 Deep Features Analysis 25
3.4.5 Subject Dependent Test 26
3.4.6 Cross Session Test 26
3.5 Conclusion and Future Work 28

4 3-Dimensional Bag of Visual Words Framework on Action Recognition 30
4.1 Overview 30
4.2 Introduction 31
4.3 Method 33
4.3.1 Network structure 33
4.3.2 3D ResNet 33
4.3.3 Feature extraction from 3D ResNet models 34
4.3.4 Single-layer classifier with sub-network nodes 34
4.4 Experiment results 37
4.4.1 Dataset 37
4.4.2 Environment setting 38
4.4.3 Extractor evaluation 38
4.4.4 Classifier evaluation 39
4.4.5 Framework evaluation 40
4.5 Conclusion 40

5 Combination of Multi-stream Features for EEG-Based Signal Emotion Recognition 42

5.1 Overview 43
5.2 Introduction 43
5.3 Method 45
5.3.1 First General Layer: Feature Extraction 45
5.3.2 Second general layer: Early fusion 47
5.3.3 Third General Layer: Classifier with Sub-network Nodes . . . 48
5.3.4 Generate Inputs to CNN Auto-encoder 48
5.4 Experiments 52
5.4.1 Dataset 52
5.4.2 Environment Setting 53
5.4.3 SEED Dataset 53
5.4.4 SEED-V dataset 56
5.5 Conclusion 59

6 Conclusion & Future Work **61**
6.1 Overview 61
6.2 Main Contributions 61
6.3 Conclusion 62
6.4 Future Work 63

Reference **64**

List of Tables

Table 3.1 Details of film clips used in the experiment. 22
Table 3.2 Network Configuration 23
Table 3.3 Mean Accuracy of Subject Dependent Test 27
Table 3.4 Mean Accuracy of Cross Session Test 27

Table 4.1 Performances of 3D ResNets 39
Table 4.2 Performances of different classifiers 39
Table 4.3 Time consumptions of different classifier (s) 40
Table 4.4 Performances of different classifiers 40

Table 5.1 Mean Accuracy of Each Stream and Multi-strem Fusion Approaches 53
Table 5.2 Mean Accuracy of Subject Dependent Test 55
Table 5.3 Mean Accuracy of Cross Session Test 57
Table 5.4 Mean Accuracy of Each Stream and Multi-strem Fusion Approaches 57
Table 5.5 Mean Accuracy of Cross Session Test 58

List of Figures

Figure 1.1 Overview of three applications. 3

Figure 2.1 Hierarchical Network with Subnetwork Nodes. 7
Figure 2.2 Classification of transfer learning. 10
Figure 2.3 Transfer learning in Deep Learning. 11

Figure 3.1 Proposed learning system from EEG data to features which is used for emotion recognition. 14
Figure 3.2 Converting EEG-based signals to matrix. 17
Figure 3.3 My proposed CNN Auto-encoder network. 19
Figure 3.4 Procedure of the stimuli playing. 22
Figure 3.5 EEG neuroScan system. 24
Figure 3.6 Comparison experiment results using different weight. 25
Figure 3.7 Deep features visualization. 26
Figure 3.8 Comparison experiment results of Subject Dependent Test. 28
Figure 3.9 Comparison experiment results of Cross Session Test. 29

Figure 4.1 Proposed framework including feature extractor and classifier. 32
Figure 4.2 Architecture of the 3D ResNet. 34
Figure 4.3 My proposed classifier. 34
Figure 4.4 The samples of video frames from UCF-101 and HMDB-51 datasets. 37

Figure 5.1 Proposed framework, 1) 1^{st} general layer: feature extraction. 2) 2^{nd} general layer: early fusion. 3) 3^{rd} general layer: classifier with subnetwork nodes. 44
Figure 5.2 Converting EEG-based signals to matrix. 48
Figure 5.3 Experiment protocol. 52

Figure 5.4 The confusion matrices of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream. 54
Figure 5.5 Comparison experiment results of Subject Dependent Test. 56
Figure 5.6 Comparison experiment results of Cross Session Test. 58
Figure 5.7 The confusion matrices of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream. 59
Figure 5.8 The visualization results of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream. 60

ACKNOWLEDGEMENTS

I would like to thank:

First of all **Dr. Yimin Yang**, for providing me with the opportunity to work under his supervision and for supporting me through all the tough times. It was not an easy journey, but I have been very fortunate to have a supervisor who cared this much about my work. His guidance, constructive suggestions, and encouragement are the reason I was able to learn and grow as a researcher. It was an absolute privilege to work with him on this research.

Dr. Ruizhong Wei, for his constructive suggestions that helped me in polishing my thesis. He provided me with many very important opinions and ideas, which is the key to the successful publication of my paper. This research would not have been possible without him.

PUBLICATIONS

Shiqi Wang, Yimin Yang, Ruizhong Wei, Q.M.Jonathan.Wu. 3D Bag of Visual Words Framework on Action Recognition. *Computers, Materials &continua (CMC)*, 2020.

Shiqi Wang, Yimin Yang, Ruizhong Wei, Q.M.Jonathan.Wu. EEG-Based Emotion Recognition Using CNN Auto-Encoder and Hierarchical Network with Subnetwork Node. *Under Review of Neurocomputing*, 2020.

Shiqi Wang, Yimin Yang, Hui Zhang, Q.M.Jonathan.Wu, Baoliang Lu. Multi-Stream features combination for EEG-Based signal Emotion Recognition. *Under Review of IEEE Transactions on congnitive and developmental systems*, 2020.

Chapter 1

Introduction

1.1 Overview 1
1.2 Motivation 3
1.3 Problem Description 4
1.4 Contribution 5

1.1 Overview

This article focuses on neural network-based feature extraction techniques and techniques. So I start with a question: What are features? Generally speaking, features are low-dimensional descriptions of training or testing objects. Features usually describe a region so that they can be distinguished. The quality of the features directly determines whether the subsequent classification and recognition will get a good result. Features should have the characteristics: repeatability, distinguishability, accuracy, effectiveness (number of features, efficiency of feature extraction), robustness (stability, invariance). The significance of feature extraction lies in choosing relevant and information-rich functions, but it can have other motivations, including: 1. general data reduction, limiting storage requirements, and improving algorithm speed; 2. reducing feature sets to save the next round of data collection or Utilize resources during the period; 3. Improve performance and improve forecast accuracy; 4. Data understanding, gain knowledge about the process of generating data or simply visualize the data.

There are many traditional feature extraction methods which are widely used, for example, Scale Invariant Feature Transformation (SIFT), Direction Gradient Histogram (HOG) and other traditional feature extraction methods. With the development of neural networks, especially deep neural networks, feature extraction based on neural networks has achieved good results. The biggest difference between deep learning and traditional pattern recognition methods is that deep learning method automatically learns features from big data, rather than features designed by hand. Good features can greatly improve the performance of the pattern recognition system. In various applications of pattern recognition in the past few decades, the features of hand-design are in the same dominant position. It mainly relies on the prior knowledge of the designer, and it is difficult to take advantage of big data. Due to the manual tuning parameters, only a few parameters are allowed in the design of the feature. Deep learning can automatically learn the representation of features from big data, which can contain thousands of parameters. Designing effective features by hand is a fairly lengthy process. Looking back at the history of computer vision development, it often takes five to ten years for a widely recognized good feature to appear. And deep learning can quickly learn new and effective feature representations from training data for new applications. Features and classification. Traditions are separated and then optimized. The neural network is a combination of both. It is possible to initially set 60 million feature parameters and then learn millions of samples. Traditional image classification, such as the recognition of local facial areas, also extracts a small area, then advances a texture feature, and then classifies.

There are many ways to extract features based on neural networks, such as extracting features through convolutional neural networks or using unsupervised learning models, auto-encoder, etc. In this thesis, I used 3D volumes and neural networks and auto-encoder to extract features. In recent studies, these models have achieved very good results.

This thesis uses three applications to test and verify the reasonable extraction of features and the positive impact of fusion on recognition results [see Fig. 1.1]. At first I chose to process the simpler 1D data, so I chose emotion recognition based on EEG signal data. I achieved the state-of-the-art result by fusing spatial domain and frequency domain features extracted by 2D CNN autoencoder and hierarchical network with sub-network. Later, to verify the effectiveness of feature learning on high-dimensional data, I choose video-based action recognition. Features Ire extracted by a pre-trained 3D ResNet, and then use a single-layer classifier with sub-network nodes to classify actions, I achieved good results. By processing 3D video data, I obtained inspiration, and the real-time domain features can improve the recognition accuracy of signals with time-domain features. Therefore, I reconstruct the time-domain features of EEG signals and fuse them to the original method and obtain better recognition accuracy. And, in order to show the generalization performance of my method, I cited EEG data sets with more classifications, and still achieved good experimental results. In the three applications I performed, video data and EEG data were processed separately. Therefore I need to prepossess those datasets. I use a variety of prepossessing methods, which are discussed in detail in chapters 3 - 5.

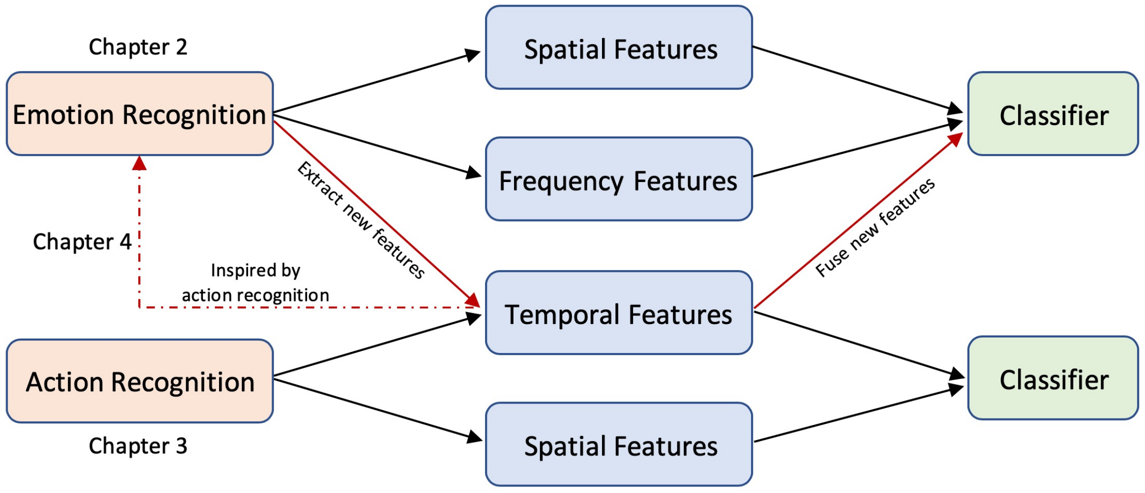

Figure 1.1: Overview of three applications.

1.2 Motivation

Computer vision is a hot research topic recently, and action recognition is an important research direction in the field of computer vision. Therefore, improving the accuracy rate of action recognition network has a great effect on the field of computer vision. Therefore, I first focused on finding ways to improve the accuracy of action recognition. I focused my research on feature extraction and transfer learning. Traditional action recognition is mainly based on 2D CNN networks, and video data contains a lot of important information and features in the time domain, so how to effectively extract these time domain features becomes extremely critical. Therefore, I first focused on finding ways to improve the accuracy of action recognition. I focused my research on feature extraction and transfer learning. Traditional action recognition is mainly based on 2D CNN networks, and video data contains a lot of important information and features in the time domain, so how to effectively extract these time domain features becomes extremely critical. Therefore, I conducted the first experiment, that is, the feature extraction and transfer learning based on the 3D CNN network described in Chapter 4. For the classifier, I chose a single-layer network with sub-network nodes. My experiments have achieved good results, so I want to verify whether my method can be applied to 1D data, because traditional CNN networks are mainly applied to 2D or 3D data. I chose EEG 1D signals as my next experimental data set. Since EEG signals are measured based on the response of the cerebral cortex, there is a spatial relationship between different electrodes. So how to use 2D CNN networks to extract spatial features is the key to my research, so I preprocess the 1D EEG data. And then use 2D auto-encoder for feature extraction. And fused with the frequency characteristics extracted through the hierarchical network with sub-network nodes, and then classified, and obtained state-of-the-art results. However, EEG data has time-domain information, so I have added extraction of time-domain features in subsequent experiments, and further improved the accuracy of classification.

1.3 Problem Description

In the three experiments carried out here, the main problems to be solved are as follows:

- The scale of video datasets is often small, so 3D CNN networks do not have enough data to train.
- The performance and effect of traditional classifiers are not very good, so I need to find a classifier with better performance and training.
- How to perform feature extraction in the frequency domain, space domain, and time domain on one-dimensional data.
- How to fuse multiple features more effectively.

1.4 Contribution

This article focuses on the extraction of focus on features. Therefore, through my experiments, this thesis has made the following contributions:

- I verified that transfer learning is effective in 3D CNN networks, and that the 3D Bag of Visual Words (BoVW) network structure has a positive effect on improving the accuracy of network recognition.
- I propose an unsupervised learning method that uses DCNN model to extract spatial features of EEG data and fuse them with those extracted through HNSN. Experimental results show that my proposed framework could provide roughly 94.05% accuracy, that is superior to other approaches.
- Based on spatial characteristics of EEG signals, I propose an EEG-based signal for the matrix algorithm, which can convert any type of features into a two-dimensional matrix.I map the EEG signals to a two-dimensional coordinate space according to the relative coordinates of the electrodes. The spatial relationship between electrodes is well expressed.
- Effect of specific channels of DE features: Previous studies [] have shown that specific channels of DE features may influence the result of EEG-based emotion recognition. Motivated by these experiments, I add weights to the 12 corresponding channel values, while I generate EEG images. After many experiments, I adjusted weight values and obtained a good performance.
- I propose a multi-stream hierarchical network framework learning behaviors of features combined from multi networks. Each stream can extract temporal, spatial and frequency features respectively, which significantly improve accuracy. I adopted an early fusion method that is better than later fusion by fusing different features into one super vector.
- Superb generalization performance. My method is evaluated on two datasets (SEED and SEED-V), I tested my framework by using original EEG signals, DE features and PSD features and compared with several state-of-the-art methods and I got the best performance.

Chapter 2

Literature Review

2.1 Overview 6
2.2 Hierarchical Network with Subnetwork Nodes 7
2.2.1 Feature Extractor 7
2.2.2 Classifier 8
2.3 Transfer Learning 9

2.1 Overview

Previous studies have proposed several state-of-the-art classifier including SVM, ELM, etc. These classifiers have very good performance in different fields. In order to more effectively improve classification accuracy and reduce training time and improve recognition speed, I adopted several methods that have been proved to be effective, including a hierarchical network with subnetwork nodes (HNSH) [51]. [5] proposed a solution for transfer learning in 3D CNN based on Kinetics video dataset, and solved the problem that during the 3D CNN training process, because the video data often has a small scale, it is not enough to train the 3D CNN network. In this chapter I will introduce HNSH network and transfer learning.

2.2 Hierarchical Network with Subnetwork Nodes

I adopted an architecture proposed by [51] called the hierarchical network with sub-network nodes (HNSN). The subnet node of this network contains a number of hidden nodes with various functions, which can be used for dimension reduction, feature extraction etc. Firstly, the network has an encoder/decoder structure, and each sub-network node can increase or decrease the dimension of data. Second, each neuron does not fully link to the output of each neuron, such as the neural representation in the mammalian cortex. Third, the output of each sub-network node is a partial feature of the input data. By merging these features, the complete features of the data can be obtained. The network consists of three parts: 1) local feature extraction 2) feature layer fusion and 3) classifier [See Fig. 2.1].

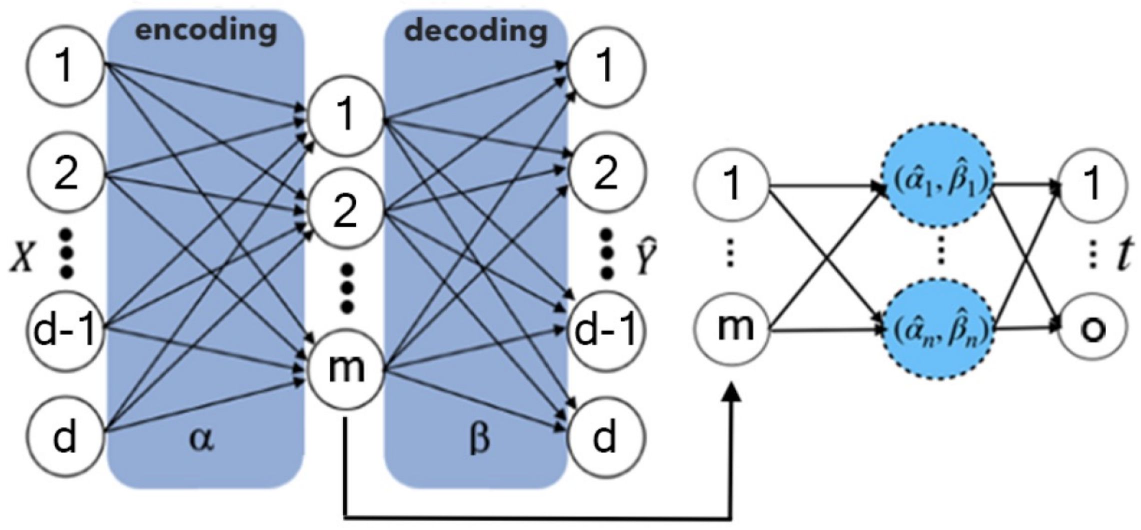
encoding
decoding
X
α
β
Ŷ
1
2
d-1
d
m
$(\hat{\alpha}_1, \hat{\beta}_1)$
$(\hat{\alpha}_n, \hat{\beta}_n)$
t
o

Figure 2.1: Hierarchical Network with Subnetwork Nodes.

2.2.1 Feature Extractor

Training a Backpropagation based autoencoder needs several iterations to adjust it's weight and bias, which consume a lot of time. Though ML-ELM reduces time consumption, hidden nodes used in the encoding layer are randomly generated that can deteriorate useful features. Instead of using several random layers, a two-layer autoencoder has been introduced, [50] where only the encoding layer weight has been generated randomly, based on which the decoding layer weight has been calculated.

The autoencoder aims is to minimize the reconstruction loss, which is the squared error between the input X and the neural network output $\hat{Y}$.

We briefly describe the Autoencoder algorithm in the following steps. *Step-1:* Randomly initialize the encoding layer weights α. After initializing the encoding layer, obtain the encoding layer output H Through Eq. 2.8.

$$\mathbf{H} = g(X\alpha + b) \tag{2.1}$$

Step-2: Calculate the decoding layer weights β through Eq. 2.9

$$\beta = \mathbf{H}^{\dagger} g^{-1}(\mathbf{Y}) \tag{2.2}$$

where $g^{-1}(\cdot)$ is the inverse function of $g(\cdot)$. For instance, if $g(\cdot) = sin(\cdot)$, then $g^{-1}(\cdot) = arcsin(\cdot)$; if $g(\cdot)$ is sigmoid function, then $g^{-1}(\cdot) = -log(1/(\cdot) - 1)$. It is worth nothing that, the pseudo inverse $\mathbf{H}^{\dagger}$ is calculated by Eq.2.10.

$$\mathbf{H}^{\dagger} = (\mathbf{H}^T\mathbf{H} + \mathbf{I}_{m\times m}/c)^{-1}\mathbf{H}^T \tag{2.3}$$

Though training time has reduced multiple times,randomized weight in the encoding layer for whole dataset increases the computation cost. Thus, it requires more memory.

2.2.2 Classifier

A network with sub-network nodes can be formed by a single hidden node or several nodes and focused on reaching the smallest training error as well as the smallest norm between the output weight and hidden nodes. Based on Bidirectional ELM [48], this network residual error will help to select an appropriate number of neurons or the sub-network itself. N numbers of distinct samples of (x_i, t_i^N) and a sigmoid or sine activation function h , the input weight and output weight of n^{th} subnetwork would be

$$\hat{\alpha_n} = h^{-1}(u(e_{n-1})) \cdot x^T(I_{d\times d}/c + xx^T)^{-1} \tag{2.4}$$

$$\hat{\beta_n} = \mathbf{H}^{\dagger} g^{-1}(en - 1) \tag{2.5}$$

where $h^{-1}(\cdot)$ has been used as the inverse function of $h(\cdot)$; u is used as normalized function which processes the input and target data by mapping from original range to (0,1]; This functions applied on the residual network error of previous subnetwork; $x^T(\mathbf{I}/c+xx^T)^{-1} = x^{-1}$ is the Moore-Penrose generalization inverse of training samples. Meanwhile, Eq. 2.10 can be used in Eq. 2.12 to get the output weight of that subnetwork. The residual error of nth subnetwork can be defined by 2.13

$$\mathbf{e}_n = t - \mathbf{h} \cdot \beta \tag{2.6}$$

2.3 Transfer Learning

With the emergence of more and more machine learning application scenarios, and the existing well-supervised learning requires a large amount of labeled data, labeling data is a boring and costly task, so transfer learning is subject to more attention.

Traditional machine learning (mainly referred to as supervised learning) is based on the same distribution assumption and requires a large amount of labeled data. However, in actual use, there may be some problems with different data sets, such as differences in data distribution, expired labeled data or expired training data, that is, it is difficult to calibrate Data will be discarded, the distribution of data in some applications will change over time. How to make full use of the previously marked data (waste utilization) while ensuring the accuracy of the model on new tasks? Based on such problems, there is research on transfer learning.

Transfer learning is a machine learning method that uses the model developed for task A as an initial point and reuses it in the process of developing the model for task B. Transfer learning is a new task to improve learning by transferring knowledge from related tasks that have been learned. Although most machine learning algorithms are designed to solve a single task, the development of algorithms that promote transfer learning is a continuous concern of the machine learning community topic of. Transfer learning is very common to humans. For example, we may find that learning to recognize apples may help to recognize pears, or learning to play the electric piano may help to learn piano. To find the similarity of the target problem, the transfer learning task is to apply the model learned in the old domain to the new domain, starting from the similarity.

Transfer learning is divided into Homogeneous transfer learning and Heterogeneous transfer learning according to the feature space. Transfer learning is divided into Inductive transfer learning, Transductive transfer learning and Unsupervised transfer learning according to the transfer scenario. Transfer learning is divided into Instance based transfer learning and Feature based transfer learning, Parameter based transfer learning, and Relation based transfer learning according to the transfer method [See Fig. 2.2].

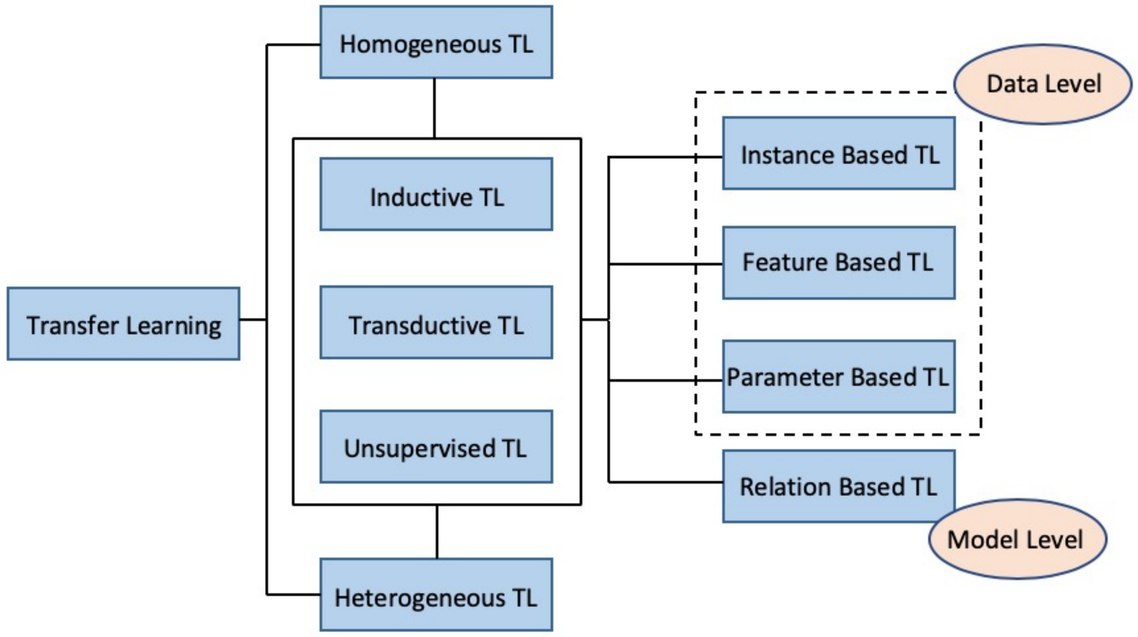

Figure 2.2: Classification of transfer learning.

The general idea of transfer learning can be summarized as: developing algorithms to maximize the use of knowledge in labeled fields to assist in the acquisition and learning of knowledge in the target field. The core of transfer learning is to find the similarity between the source domain and the target domain and use it reasonably. With this similarity, the next step is how to measure and use this similarity. The goal of the measurement work has two points: One is to measure the similarity of the two fields, not only tell us qualitatively whether they are similar, but also give the degree of similarity more quantitatively. The second is to take the metric as the criterion, and increase the similarity between the two fields through the learning methods we want to use to complete the transfer learning.

The most useful occasion for transfer learning is if you try to optimize the performance of task B, usually this task has relatively little data. For example, in the radiology department, you know that it is difficult to collect a lot of radiographic scans to build a radiology diagnostic system with good performance, so in this case, you may find a related but different task, such as image recognition, in which you It may have been trained with 1 million pictures and learned a lot of low-level features, so that may help the network to do better in the task of radiology, although the task does not have so much data.

If the difference between the two fields is particularly large, transfer learning cannot be used directly, because in this case the effect is not very good. In this case, the above method is recommended to migrate step by step between two domains with low similarity

In deep learning, deep learning is often combined with transfer learning. Deep learning requires a large amount of high-quality annotated data. Pre-training + fine-tuning is a very popular trick in deep learning, especially in the field of images. Many times, pre-trained ImageNet will be selected to initialize the model [See Fig. 2.3].

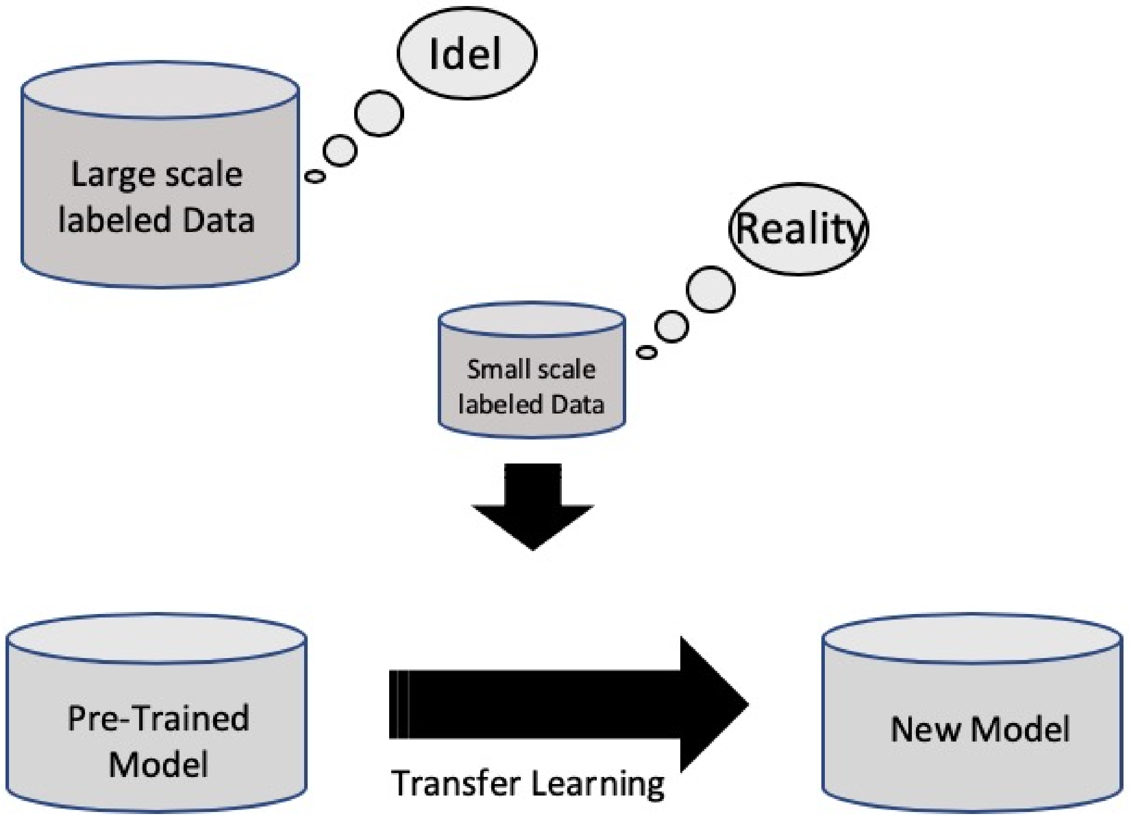

Figure 2.3: Transfer learning in Deep Learning.

Chapter 3

EEG-Based Emotion Recognition Using CNN Auto-Encoder and Hierarchical Network with Subnetwork Node

3.1 Overview 13
3.2 Introduction 13
3.3 Method 15
3.3.1 Differential Entropy Features 15
3.3.2 Making Images from EEG Features 16
3.3.3 CNN Auto-Encoder with ResBlock 18
3.3.4 Hierarchical Network with Subnetwork Nodes 20
3.3.5 Feature Fusion 20
3.4 Experiments 21
3.4.1 Dataset 21
3.4.2 Environment Setting 23
3.4.3 Weight Coefficient Test 24
3.4.4 Deep Features Analysis 25
3.4.5 Subject Dependent Test 26
3.4.6 Cross Session Test 26
3.5 Conclusion and Future Work 28

3.1 Overview

One-dimensional data is the most common form of data in my lives, and the processing of one-dimensional data is also simpler than that of high-dimensional data. Therefore, we chose to process one-dimensional data as my first experiment. Emotion is one of the peculiar characteristics of human beings; it affects human behavior. Therefore, in this chapter we choose emotion recognition based on one-dimensional EEG signals, and we proposes a new method of emotion recognition, which uses convolutional neural network (CNN) auto-encoder and hierarchical network with sub-network node (HNSN) to extract features and classify emotions. Using the convolutional neural network to process electroencephalography (EEG) signal matrix can fully consider the spatial relationship of the EEG signal, such as the active and inactive areas of a certain emotion in the cerebral cortex. Combining these CNN auto-encoder deep features with features generated from HNSN, the network can give a more reliable cognition. After experimenting with the proposed method, it was found that compared with other methods, the proposed method gives better results.

3.2 Introduction

Emotion recognition has always been a hot research topic in the past three decades. It is the process of identifying human emotion, most typically from facial expressions, as well as from verbal expressions, but also from bio-signals, including electrocardiogram, ocular electricity, myoelectricity and EEG. Compared with these signals, EEG is widely used for its high accuracy and stability [27]. Emotional recognition based on EEG can be traced back to 1985 [1]. In the past 20 years, with the rapid development of machine learning, the use of machine learning to identify human emotions has achieved good results [20].

The most prominent methods are based on statistics, waves and fusions. Machine learning methods have been proposed for emotion recognition. The early classification methods (single-layer feed-forward network, fuzzy k-means and support vector machine (SVM)) achieved moderate results in experiments with multiple emotional states [12] (2 intermediate or above). For example, [24] used the f-score index, based on the ratio of emotion recognition between and within classes. By classifying four emotion states at 26 electrodes, they achieved an average accuracy of 82.29%. [7] used time frequency data to identify three emotions, obtaining 63% accuracy, and fused different features and samples, achieving 80% mean accuracy. [53] preprocessed EEG signals by differential entropy (DE) [9] method, grouped all electrodes, selected the characteristic data of 12 electrodes and classified them by SVM. They obtained best accuracy of 86.65%. Their experimental results show that the corresponding channel features affect the accuracy of emotion recognition. Later, [51] proposed a hierarchical network with sub-network nodes and got mean accuracy of 91.51% on DE features from full channels.

Although the above research uses the traditional shallow model as the classifier, the deep learning method has recently been introduced in emotion recognition based on EEG data. Deep belief network (DBFs) is used to analyze human emotions. For example, [55] used DE features to train a DBF and achieved a mean accuracy of 87.62%. Unsupervised learning methods, for example, auto-encoder, are also used for extracting deep features from EEG signals. The limitation of previous studies is that

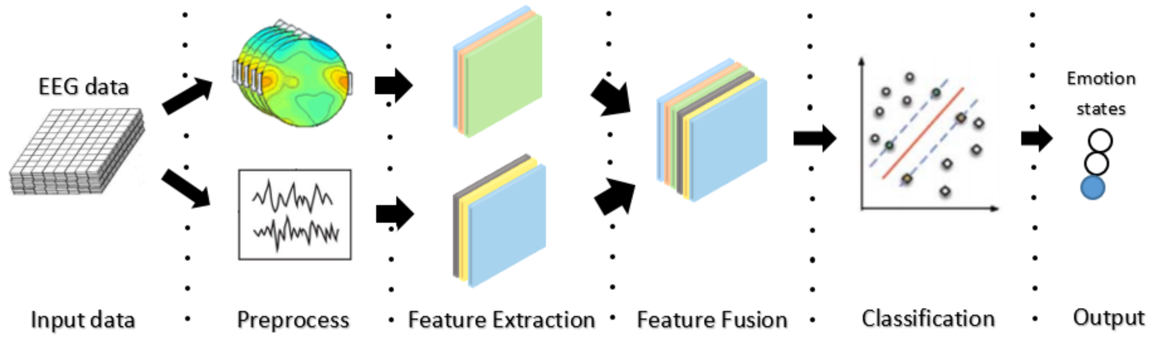

Figure 3.1: Proposed learning system from EEG data to features which is used for emotion recognition.

they did not consider the spatial connections between electrodes. In the neuroscience field, EEG signals usually contain five frequency bands, such as delta (δ: 1–3 Hz), theta theta (θ: 4–7 Hz), alpha (α: 8–13 Hz), beta (β: 14–30 Hz), and gamma (γ: 31–50 Hz) [28] to examine their relationship with the emotional states. Alpha power asymmetry is a common indicator for evaluating emotional states, and there are spectral differences in asymmetric pairs of electrodes in the anterior regions of the brain [19]. Emotion states were also associated with spectral changes in other parts of the cerebral cortex, such as right parietal alpha changes, theta power changes in the right parietal [33], frontal-midline (FM) theta power [32], power asymmetry in the beta-parietal region, and gamma spectrum changes in the right parietal region [3]. Motivated by these limitations, this chapter proposed an unsupervised deep convoluational neural network (DCNN) method that combines with HNSN for EEG-based emotion recognition [see Fig. 3.1]. Particularly, this chapter makes the following contributions.

- I propose an unsupervised learning method that uses DCNN model to extract spatial features of EEG data and fuse them with those extracted through HNSN. Experimental results show that my proposed framework could provide roughly 94.05% accuracy, that is superior to other approaches.
- Based on spatial characteristics of EEG signals, we propose an EEG-based signal for the matrix algorithm, which can convert any type of features into a two-dimensional matrix. I map the EEG signals to a two-dimensional coordinate space according to the relative coordinates of the electrodes. The spatial relationship between electrodes is well expressed.
- Effect of specific channels of DE features: Previous studies [9, 27, 53] have shown that specific channels of DE features may influence the result of EEG-based emotion recognition. Motivated by these experiments, we add weights to the 12 corresponding channel values, while we generate EEG images. After many experiments, we adjusted weight values and obtained a good performance.

3.3 Method

3.3.1 Differential Entropy Features

[9] proposed the effective features called differential entropy (DE) extend the concept of Shannon entropy and are used to measure the complexity of continuous random variables. Since EEG data has high low-frequency energy in high-frequency energy, DE has the ability to distinguish the EEG mode between low-frequency and high-frequency energy. This is the first time that [9] introduced EEG-based emotion recognition.

The original calculation formula of differential entropy is defined as

$$H(x) = -\int_x f(x)\log(f(x))dx \tag{3.1}$$

If the time series X obeys the Gauss distribution N(μ,δ), the DE features can be obtained by

$$\begin{aligned} H(x) &= -\int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\delta^2}} e^{-\frac{(x-\mu)^2}{2\delta^2}} \log\left(\frac{1}{\sqrt{2\pi\delta^2}} e^{-\frac{(x-\mu)^2}{2\delta^2}}\right) dx \\ &= \frac{1}{2}\log 2\pi\delta^2 \end{aligned} \tag{3.2}$$

where the time series x obeys the Gauss distribution $N(\mu,\ \sigma^2)$. Experiments show that for fixed-length EEG signal sequences,, DE is equivalent to the logarithm energy spectrum in a certain frequency band [36]. I use this method to extract the DE features of 5 corresponding frequency bands.

DE features can be converted to 5 different frequency bands (delta: 1-3 Hz, theta: 4-7 Hz, alpha: 8-13 Hz, beta: 14-30 Hz, gamma: 31-50 Hz) with time complexity $O(K\ N\ \log\ N)$ where N is the size of samples, and K is the number of electrodes.

To test the generalization performance of my architecture, we also extracted traditional power spectral density (PSD) features. For both the DE and PSD features, we use the linear dynamic system (LDS) method to further filter out irrelevant components, and consider the temporal dynamics of emotional state [37].

3.3.2 Making Images from EEG Features

EEG signals are similar to other signals in time and frequency domains, but the most prominent features are in frequency domain. EEG data are collected at different spatial locations in the cerebral cortex and consists of multiple time series. However, as has been pointed out, EEG signals have additional spatial dimensions.

The one-dimensional data obtained by analyzing this measurement method is the standard method in EEG data analysis. However, this approach obviously ignores the relationship between space and frequency. Therefore, we convert the measured values into a two-dimensional matrix to preserve the spatial structure and use multiple channels to represent the spectrum of bands with different frequencies.

Both DE and power spectral density (PSD) features contains the values of 62 electrodes in five frequency bands. The detailed order of the channels is included in the dataset. The EEG cap according to the international 10-20 system for 62 channels and based on previous studies [9, 53], 12 channels [FT7, FT8, T7, T8, C5, C6, TP7, TP8, CP5, CP6, P7 and P8] are selected in my experiment.

Filters in convouational neural networks (CNNs) can learn patterns of adjacent

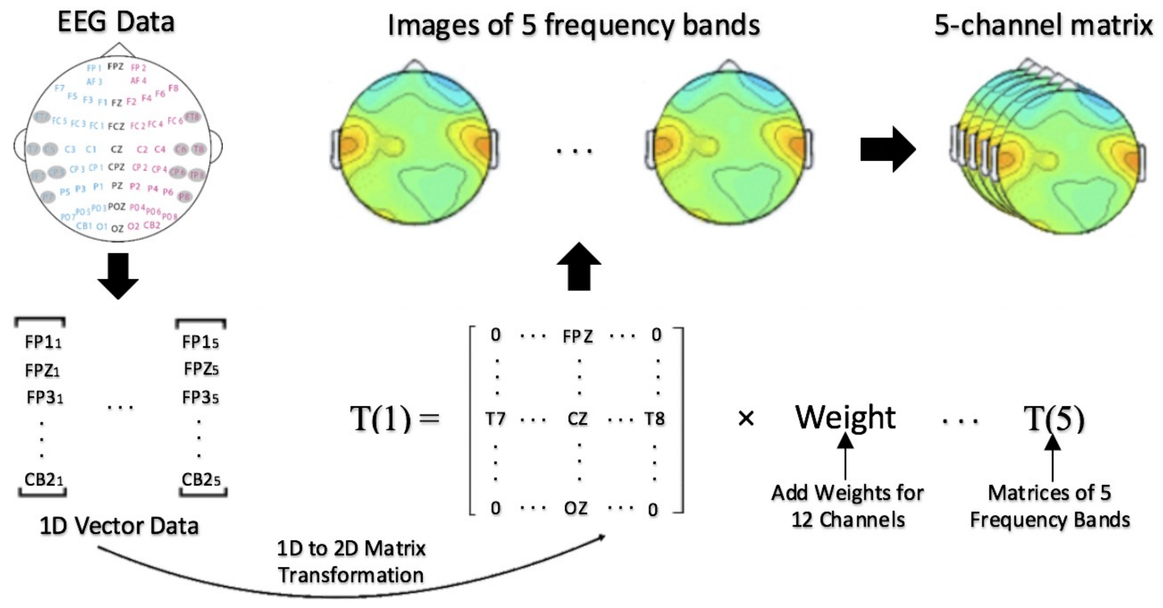

Figure 3.2: Converting EEG-based signals to matrix.

values in the connectivity matrix, which enables CNN to learn spatial features of a two-dimensional matrix. In order to generate the input matrix data of CNN, we need to convert the EEG feature data as two-dimensional matrices. I converted the EEG data into a 9×9 matrix, for the boundaries of the matrix and positions of the null electrodes are padding with zeros. The matrix can be defined as

$$M(n) = \begin{bmatrix} 0 & 0 & 0 & FP1 & FPZ & FP2 & 0 & 0 & 0 \\ 0 & 0 & 0 & AF3 & 0 & AF4 & 0 & 0 & 0 \\ F7 & F5 & F3 & F1 & FZ & F2 & F4 & F6 & F8 \\ FT7 & FC5 & FC3 & FC1 & FCZ & FC2 & FC4 & FC6 & FT8 \\ T7 & C5 & C3 & C1 & CZ & C2 & C4 & C6 & T8 \\ TP7 & CP5 & CP3 & CP1 & CPZ & CP2 & CP4 & CP6 & TP8 \\ P7 & P5 & P3 & P1 & PZ & P2 & P4 & P6 & P8 \\ 0 & PO7 & PO5 & PO3 & POZ & PO4 & PO6 & PO8 & 0 \\ 0 & 0 & CB1 & O1 & OZ & O2 & CB2 & 0 & 0 \end{bmatrix} \tag{3.3}$$

Where n represents frequency bands. Previous studies [9, 27, 53] have proved that 12 selected channels have a positive impact on emotion recognition. In order to increase the impact of the 12 special electrodes on the EEG signal analysis, we added weights to the values of these corresponding 12 electrodes. These weights are adjustable, and the weights of these 12 channels are equal.

$$Weight = \begin{bmatrix} & & & \cdots & & \\ W & & & & & W \\ W & W & & & W & W \\ W & W & & & W & W \\ W & & & & & W \\ & & & \cdots & & \end{bmatrix} \tag{3.4}$$

The weight is also a two-dimensional matrix and has the same dimension as the EEG data matrix. W in the matrix represents the weight of the corresponding electrode data. Here, we totally have 12 weights to 12 corresponding electrodes in total. I multiply the EEG matrix data by the weight

$$T(n) = M(n) \times Weight \tag{3.5}$$

So the EEG data meshes D is created as follows:

$$D(n) = [T_1, T_2, ..., T_5] \tag{3.6}$$

In a later section of the experiment, we will list the data of the influence of weights on the accuracy of emotional recognition. I used this method for five different frequency bands to get five two-dimensional matrices corresponding to different frequency bands. The five two-dimensional matrices are then combined to form a 2D matrix with five channels [See Fig. 5.2] with the size of $9 \times 9 \times 5$.

3.3.3 CNN Auto-Encoder with ResBlock

Encoder decoder network is a symmetric CNN structure. the input data is gradually reduced in dimension, converted to a feature map with a smaller space and more channels (encoders), and then converted back to the input shape (decoders). In the auto-encoder, the encoder part is composed of multiple convolution layers, which usually have filters with different steps, and the decoder part is composed of a series of deconvolution layers or by adjusting the size. To add further depth, an additional convolution layer is usually inserted after each layer. Skip connections between corresponding feature maps are widely used to combine different levels of information, which can effectively enhance the transmission of features and are also conducive to gradient propagation and accelerated convergence. Additional convolution layers are usually inserted after each layer to further increase the depth.

Previous experiments [29, 43] have shown that the encoder and decoder structure is effective, especially in the field of image processing. However, considering the following factors, the direct use of encoder-decoder structure in my experiment cannot achieve the best results.

First of all, filters in CNNs try to learn patterns from a given dataset, a larger area can better learn the EEG signal perception of spatial relationships, however, larger area awareness can lead to more encoder decoder module stack up, because many elements between channels rapidly increase the number of parameters, and greatly increase the computational complexity, and second, the middle-figure features of the space are too small and are not a very good reserve for the space information signal. Finally, adding more convolution layer drops to the encoder and decoder module will slow down the network convergence.

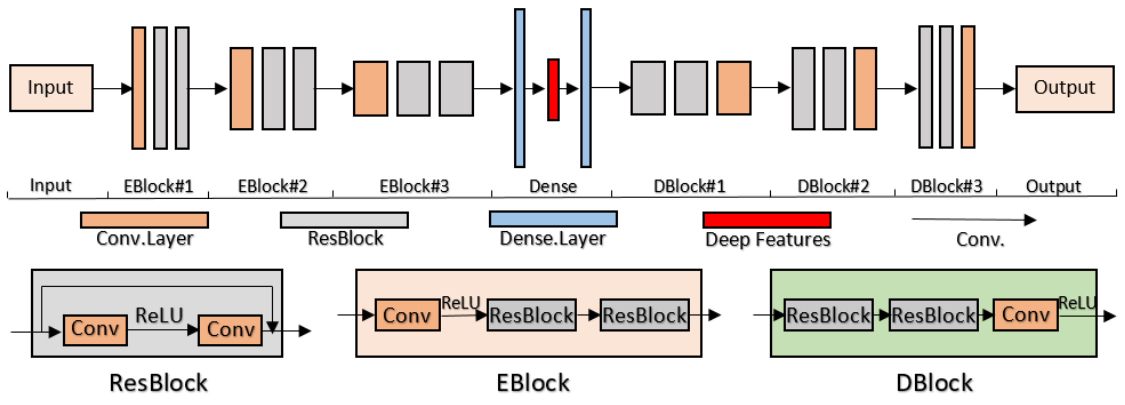

Figure 3.3: My proposed CNN Auto-encoder network.

I have made some modifications to adapt the auto-decoder network to my framework. First, we tried the shallow autoencoder, which has positively improved my experimental results. As the number of network layers increases, my method performance also improves. However, as the number of network layers increases, the network convergence is slow and the stability was getting worse. In order to improve my encoder/decoder network structure, we introduce the residual learning block [14]. According to [29] and my experimental results, the original building blocks in ResNet [14] (without batch normalization) could not bring about a better structure, so we chose to use ResBlock. As shown in Fig. 3.3, the encoder we used encoder ResBlocks (EBlocks) which contains a convolution layer followed by several ResBlocks. Each of the following ResBlocks contains two convolutional layers. Several dense layers followed by the last convolutional layer of encoder part reduce the feature dimensions greatly. Decoder part is the mirror of the encoder part. So, we extract deep features from the last dense layer of the encoder part. The modified network is expressed as

$$\begin{aligned} e &= Net_E(I), \\ f &= Dense_E(e), \\ g &= Dense_D(f), \\ D &= Net_D(g), \end{aligned} \tag{3.7}$$

where, Net_E and Net_D are encoder and decoder CNNs, and $Dense_E$ and $Dense_D$ are encoder and decoder dense layers. EBlocks and DBlocks are used in Net_E and Net_D, respectively. The value f represents deep features, used to do the classification.

My network contains total 45 layers. There are three EBlocks, two dense layers in my encoder, and three DBlocks, one dense layer in my decoder. The number of kernels is 32, 64 and 128 in EBlocks. For DBlocks, they are 128, 64 and 32. The first dense layer has 1024 units, while the second one has 500 units, meaning that my extracted deep features have the shape of $n \times 500$ (n is the number of samples). I also tested different number of nodes of the second dense layer. While the number of 500, my classifier achieved the best performance. I use the activation function rectified linear units (ReLU), and all kernels are set to size five.

3.3.4 Hierarchical Network with Subnetwork Nodes

I adopted an architecture proposed by [51] called the hierarchical network with subnetwork nodes (HNSN). The subnet node of this network contains a number of hidden nodes with various functions, which can be used for dimension reduction, feature extraction etc. Firstly, the network has an encoder/decoder structure, and each subnetwork node can increase or decrease the dimension of data. Second, each neuron does not fully link to the output of each neuron, such as the neural representation in the mammalian cortex. Third, the output of each sub-network node is a partial feature of the input data. By merging these features, the complete features of the data can be obtained. The network consists of three parts: 1) local feature extraction 2) feature layer fusion and 3) classifier.

3.3.5 Feature Fusion

I extracted the spatial and frequency characteristics of EEG data in my experiment, and we needed to find a way to fuse those two features. [35] and [52] shows that early fusion (by simply combining different features into super-vectors) can achieve better results if the data contains correction information. For example, we have two different features extracted from two different networks. I define those features extracted from the first network as $\mathbf{H}^1 = \left\{\mathbf{H}_1^1, \mathbf{H}_2^1, ..., \mathbf{H}_c^1\right\}$, and features extracted from the second network as $\mathbf{H}^2 = \left\{\mathbf{H}_1^2, \mathbf{H}_2^2, ..., \mathbf{H}_c^2\right\}$. After early fusion, we can generate composite features.

$$\mathbf{H}^{1\bigoplus 2} = [\mathbf{H}_1^1, \mathbf{H}_2^1, ..., \mathbf{H}_c^1, \mathbf{H}_1^2, \mathbf{H}_2^2, ..., \mathbf{H}_c^2]^T \tag{3.8}$$

Given that several features $\mathbf{H}^1, ..., \mathbf{H}^c$, K represent combinatorial operators, combinational features can be expressed as

$$\begin{aligned} \mathbf{H}^{1\bigoplus 2} &= K(\mathbf{H}^1, \mathbf{H}^2), \\ \mathbf{H}^{1\bigoplus 2\bigoplus 3} &= K(K(\mathbf{H}^1, \mathbf{H}^2), \mathbf{H}^3), \\ &\vdots \\ \mathbf{H}^{1\bigoplus 2\cdots\bigoplus c} &= K(\ldots K(K(\mathbf{H}^1, \mathbf{H}^2), \mathbf{H}^3)\ldots). \end{aligned} \tag{3.9}$$

In the field of biology, hybrid neurons play an important role in the realization and coding of brain functions [31]. The brain screens the sub-space features, produced by neurons to remove relevant factors. However, at the same time, the brain reconstructs sub-space features and generates complex and stable behaviors. Similar to brain behavior, spatial feature dimensions decrease gradually in the layered HNSN architecture. Finally, the training data is used for training classifier by early fusion method.

My proposed framework supports the extraction and combination of arbitrary types of data and has multiple capabilities, which we will summarize in later sections.

3.4 Experiments

3.4.1 Dataset

Previous studies [9, 55] have shown that raising human emotions through movie clips [see Table. 3.1 and Fig. 3.4] is reliable. In this chapter, we adopted the SEED dataset [53], one of the largest databases, which has been popularly used for EEG-based emo-

No.	Emotion label	Film clips sources
1	negative	Taangshan Earthquake
2	negative	Back to 1942
3	positive	Lost in Tailand
4	positive	Flirting Scholar
5	positive	Just Another Pandora's Box
6	neutral	World Heritage in China

Table 3.1: Details of film clips used in the experiment.

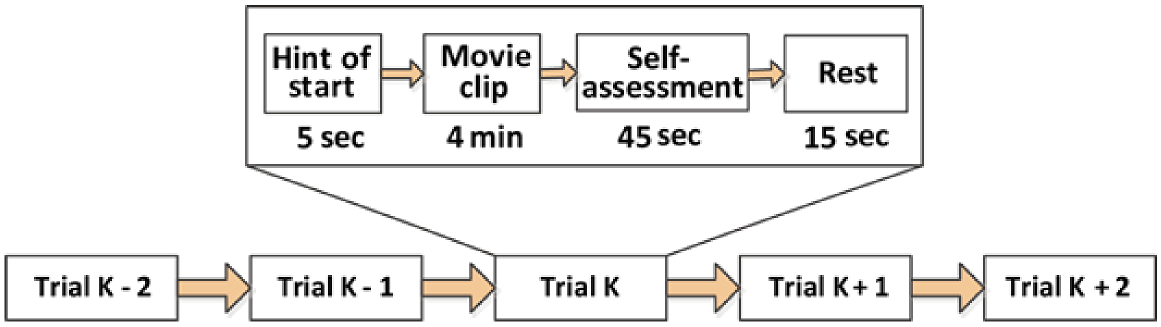

Figure 3.4: Procedure of the stimuli playing.

tion recognition. 15 Chinese film clips were chosen as stimuli used in the experiments. Each film clip is about four minutes long, and carefully selected important clips enable it to create coherent emotions, which can well trigger the corresponding human emotions. There are 15 clips. Each clip has five seconds of prompts before and 45 seconds of feedback after each clip. All movie clips are sorted according to different emotions to ensure that the same emotional movie clips are displayed discontinuously. In order to ensure the accuracy of emotional records, each participant completed the questionnaire immediately after watching. In order to eliminate the influence of gender on emotional recognition, seven males and seven females (a total of 14 subjects) participated in data collection. Each participant added three data acquisitions at different times to test the temporal stability of emotion. So we had a total of 3×14 experimental data. All movie clips are divided into three emotions (positive, neutral and negative). Each emotional data is divided into five different frequency bands, and DE and PSD features are extracted [9]. According to the international 10-20 system, the EEG NeuroScan system Fig. 3.5 was used to record EEG signals at a

Table 3.2: Network Configuration

Methods	parameter details
SVM	Linear Kernel, search space $2^{[-10:10]}$ with a step of one.
KNN	Baseline k equals 5.
ELM	1000 hidden neurons, search space $2^{[-10:10]}$ with a step of one.
H-ELM	N1=N2=300, N3=1000, search space for C1 and C2 is $2^{[-10:10]}$ with a step of one.
GELM	the number of hidden layer neurons is fixed as 10 times of the dimensions of input and we adopt a 5-fold cross-validation scheme.
DBN	first and second layer of DBN is selected from the ranges of [200:200] and [150:500], respectively.
HNSN	search space for C1 and C2 is $2^{[-10:10]}$ with a step of one. Three subnetwork nodes.In each subnetwork node, 500 hidden nodes are used.
OURS	CNN part: we use 100 epochs, learning rate of first 30 epochs is 0.1, from 30 to 70 epochs we use 0.01, from 70 to 100 we use 0.01. HNSN part: search space for C1 and C2 is $2^{[-10:10]}$ with a step of one. Three subnetwork nodes.In each subnetwork node, 500 hidden nodes are used.

sampling rate of 1000 Hz. There are 62 active AgCl electrode channels for recording EEG signals. The impedance of each channel in the cap was controlled to less than 5 KΩ.

3.4.2 Environment Setting

In this section, we systematically compare the performance of nine methods for emotion recognition: 1) SVM; 2) Extreme Learning Machine (ELM); 3) Graph regularized Extreme Learning Machine (GELM) [54]; 4) Hierarchical ELM (H-ELM) [42]; 5) DBNs [53]; 6) Bimodal Deep Auto-Encoder (BDAE) [22]; 7) HNSN [51]; and 8) Logistic regression (LR); 9) the proposed method. These methods use the above two features (DE and PSD) as input. For SVM and ELM, parameters are set up from space [2^{-10}, 2^{-9},..., 2^{10}] in each experiment. For GELM, we set the number of hidden layer neurons to 10 times the input data dimension, and adopted the cross-validation scheme [54]. For H-ELM, we used 300 hidden neurons in the first and second layers, and 1000 hidden neurons in the third layer. ($N1 = N2 = 300$, $N3 = 1000$). In DBN, two hidden layers are used. I trained 1000 epochs with batch size of 201. I set unsupervised and supervised learning rates of 0.1 and 0.5. In each experiment, the number of neurons in the first layer of DBN network was selected from [200 : 500],

Figure 3.5: EEG neuroScan system.

and the number of neurons in the second layer was selected from the range of [150 : 500]. For HNSN, to compare fairly with ELM/SVM, we selected the same values of C1 and C2, which all choose regularization parameters from space $[2^{-10}, 2^{-9}, ..., 2^{10}]$. For my method, in CNN auto-encoder part, we use epoch 100 and learning rate equals to 0.1 for first 30 epochs, 0.01 for 30 to 70 epochs and 0.001 for rest epochs, we chose the batch size of 128, and use SGD optimizer, totally trained 100 epochs. For HNSN part, in order to be consistent with other methods mentioned above, we choose the same parameters C1 and C2. My proposed method includes early fusion. Table 3.2 shows the specific settings for different methods.

3.4.3 Weight Coefficient Test

In my approach, weights are added to 12 channels. In order to prove the influence of weight factors on emotion recognition, we conducted the following experiment. I selected multiple different weights and performed multiple experiments and compared the test results. I selected weights from the range of [1:1.4]; when the weight is 1, it means that no weight is added. It is obvious from the experimental results that when

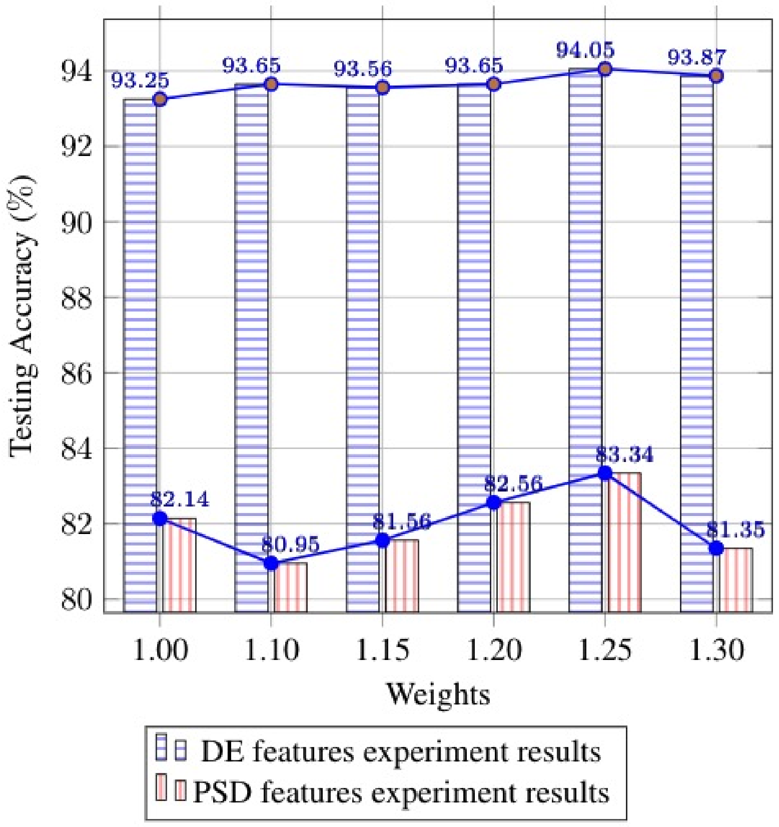

Figure 3.6: Comparison experiment results using different weight.

the weight is 1.25, we get the highest accuracy [see Fig. 3.6].

3.4.4 Deep Features Analysis

I used a weight of 1.25 to generate EEG matrices and train the CNN network. Deep features are extracted from the last dense layer of CNN encoder part, which has the shape of 1 × 500 (that layer has 500 neurons). To see benefits of deep features, extracted by my proposed method, we run a t-distributed stochastic neighboring embedding algorithm (t-SNE) to find two-dimensional embeds of high-dimensional feature space and plot them as colored points, according to semantic categories in a particular hierarchy, as shown in Fig. 3.7.

Data from each emotion can be gathered in the potential space, and the generated data is close to the corresponding actual data, meaning that the generated data reflects enough realistic information. The generated data complements the training manifold, bringing better margin for the classifier.

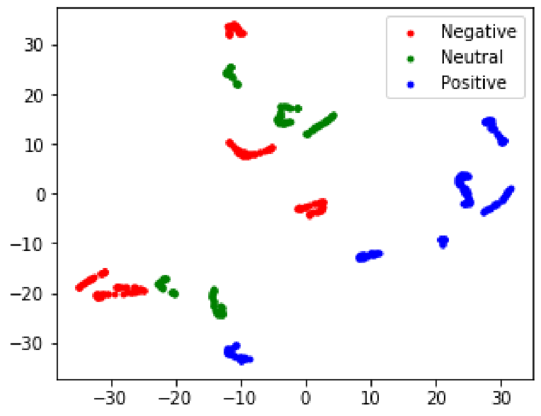

Figure 3.7: Deep features visualization.

3.4.5 Subject Dependent Test

Subject dependence means use a person's emotional responses, stimulated from different film clips to predict this person's emotions. In my dataset, we have total 14 subjects' data in three time periods or sessions. So in my experiment, we choose first nine EEG data from each session as training data, while the testing data is from the last six sessions. The training and testing data are from different sessions of the same subject. For consistency with other methods, we used only the first and last sessions in the SEED dataset.

I compared the accuracy of my proposed methods with SVM, ELM, H-ELM, DBNs [53], LR, Adaptive Subspace Feature Matching (ASFM) [6], Dynamical Convolutional Neural Networks (DGCNN) [39], BDAE [22], GELM [54] and HNSN [51] to show the advantages of my proposed method. Table 5.2 and Fig. 5.5 show the results of comparative experiments. From the table, we can see that the accuracy of my proposed method is significantly higher than that of other methods. In addition, my results are consistent with previous studies [53] and [9]. In EEG-based emotional recognition, DE features have the best performance.

3.4.6 Cross Session Test

Cross session refers to the prediction of a person's emotions through the corresponding emotions of the same person at different times. This tests the stability of emotion recognition model in time domain. As we know that we have total 14 subjects' data in three time periods or sessions, there is a week or longer interval between each

Methods	DE	PSD
SVM	89.99	59.60
ELM	82.92	60.80
LR	82.70	-
DBNs [53]	86.08	61.90
ASFM [6]	83.51	-
DGCNN [39]	90.40	81.73
BDAE [22]	91.01	85.10
GELM [54]	91.07	72.75
HNSN [51]	91.51	73.81
OURS	**94.05**	**83.34**

Table 3.3: Mean Accuracy of Subject Dependent Test

session. This is the novelty of SEED that we developed, compared to my existing emotional EEG data set. By doing cross session test, my goal is to assess whether the performance of my emotional recognition model is stable over time. For the dataset, each subject has three sessions, conducted on different dates. In this test, we use the first two sessions as input to train my network, and last session is used for testing data. For all the other methods, the way of parameter selection is the same as mentioned in the previous session.

Methods	DE	PSD
SVM	60.95	48.10
ELM	77.62	62.86
H-ELM	80.67	59.05
DBNs [53]	76.57	62.98
ASFM [6]	84.64	-
DGCNN [39]	79.95	64.27
BDAE [22]	66.08	66.23
GELM [54]	79.28	-
HNSN [51]	80.84	61.43
OURS	**88.45**	**68.21**

Table 3.4: Mean Accuracy of Cross Session Test

I compared the accuracy of my proposed methods with ELM, H-ELM, SVM, DBN, ASFM, DGCNN, BDAE, GELM, HNSN to show the advantages of my proposed method. Table 5.5 and Fig. 5.6 show the performance evaluation of the proposed method and other methods. As can be seen from Figure 7, the profit of the proposed

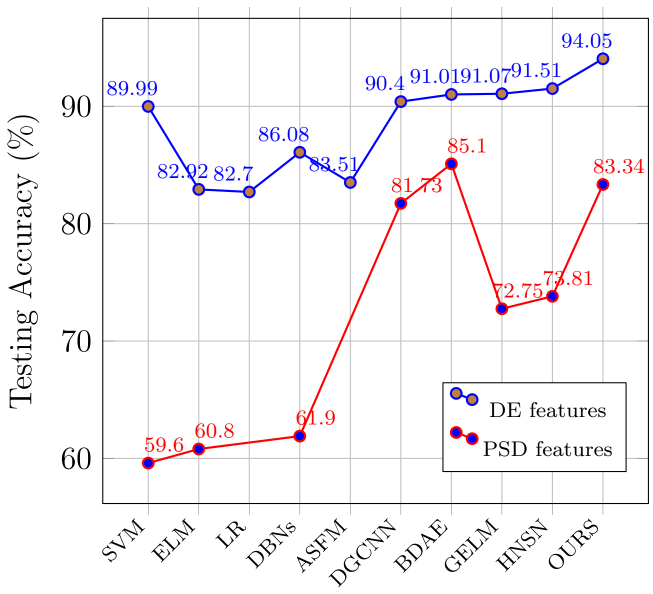

Figure 3.8: Comparison experiment results of Subject Dependent Test.

test accuracy method is obvious.

The results suggest that the relationship between changes in emotional state and EEG signals is stable for a person over time. The stability of EEG signal decreases with time. The stability of EEG signals in time domain needs to be studied in the future.

3.5 Conclusion and Future Work

This chapter presents a CNN auto-encoder combined with a hierarchical network with sub-network nodes for EEG-based emotion recognition. I can get the following three conclusions: 1) Convert EEG-based signals to images with spatial location weight makes features stronger and 2) features extracted from both nonlinear and linear multi-layer network, rather than extracted from a single linear network and 3) we use early fusion to combine multiple features. The experimental results show that my method is suitable for all kinds of EEG data and has good performance. In the future, in order to test the effect of cerebral cortex on emotion recognition, we intend to add different weights to EEG data in different frequency bands and select different electrodes. Currently the classifier we use is not online sequential; we plan to improve my network structure to make it online sequential, which can be trained by several epochs and the network performance will be significantly improved.

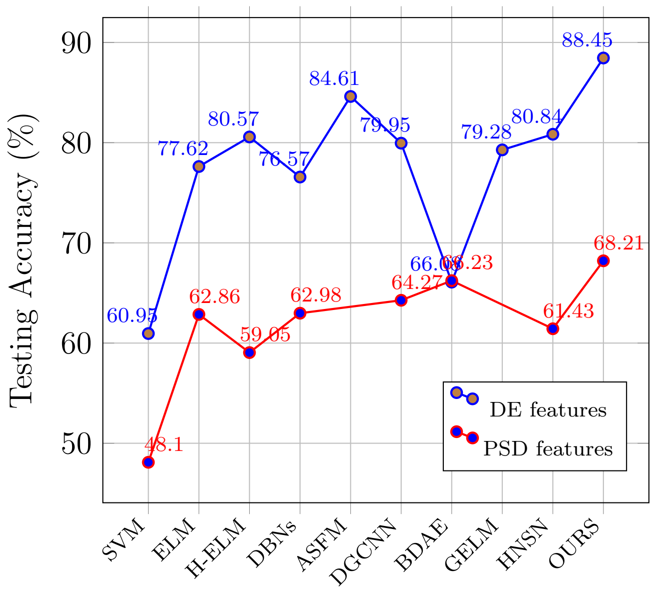

Figure 3.9: Comparison experiment results of Cross Session Test.

Chapter 4

3-Dimensional Bag of Visual Words Framework on Action Recognition

4.1 Overview 30
4.2 Introduction 31
4.3 Method 33
4.3.1 Network structure 33
4.3.2 3D ResNet 33
4.3.3 Feature extraction from 3D ResNet models 34
4.3.4 Single-layer classifier with sub-network nodes 34
4.4 Experiment results 37
4.4.1 Dataset 37
4.4.2 Environment setting 38
4.4.3 Extractor evaluation 38
4.4.4 Classifier evaluation 39
4.4.5 Framework evaluation 40
4.5 Conclusion 40

4.1 Overview

In the previous chapter we processed one-dimensional EEG signals, so in this chapter we chose to process 3D data, namely video, to verify the effect of features on the recognition results. Due to human action recognition plays a crucial role in the video analysis framework. However, a given video may contain a variety of noises, such as an unstable background and redundant actions, completely different from the key actions. These noises pose a great challenge to human motion recognition. So, in order to show the accuracy of feature learning for 3D data recognition, we propose a new method based on the 3D Bag of Visual Words (BoVW) framework. My method includes two parts: The first part is the video action feature extractor, which can identify key actions by analyzing action features. In the video action encoder, by analyzing the action characteristics of a given video, we use the deep 3D CNN pre-trained model to obtain expressive coding information. A classifier with sub-network nodes is used for the final classification. The extensive experiments demonstrate that my method leads to an impressive effect on complex video analysis. My approach achieves good performance on the datasets of UCF101 (85.3%) and HMDB51 (54.5%).

4.2 Introduction

Video action recognition is the basic building block in various applications such as video retrieval, natural human-machine interaction, video surveillance, and digital entertainment [16, 25, 40]. In action recognition, there are two important and complementary aspects: appearance and dynamics. Video often has some complex factors, such as camera motion, scale change and viewpoint change. Therefore, whether the action recognition system can extract and utilize the relevant feature information is the key to its performance. However, it is not easy to extract features effectively. Therefore, the question of how to design a network structure to deal with these problems and retain classified information becomes crucial.

The recent rise of recurrent neural networks (RNNs) have been successfully applied to action recognition [8, 22]. Existing 3D motion recognition methods based on RNN are mainly used for time-domain modeling of long-term context information, representing the dynamic based on motion. However, in the spatial domain, there are also strong dependencies between nodes. For 3D action recognition tasks, the spatial configuration of nodes in video frames may be very recognizable. [26] proposed a spatial-temporal long short-term memory (ST-LSTM) network and achieved a good performance.

The BoVW framework has recently been used in motion recognition with good results. This framework includes two parts, feature extractor and classifier. Most of the BoVW models adopt Fisher vectors of improved dense trajectories [11, 46] or CNN features [47] with a classifier such as support vector machine (SVM), and achieve reliable results on pre-segmented video datasets, such as UCF-101 [41] and HMDB51 [21].

In action recognition field, 3D CNNs have recently been more effective than the CNNs with two-dimensional (2D) kernels [5]. Recently, 3D CNNs has been used in accurate action recognition. However, the 2D model still has strong associations with video data. Even well-organized 2D models [44, 45] cannot overcome the advantages of 2D CNNs combining stacked flow with RGB images [38], mainly because video datasets usually have small data-scales, preventing optimization of a large number of parameters in 3D CNNs cannot be optimized. In addition, 3D CNNs can only be trained on a video data set from scratch, while 2D CNNs can be pre-trained on ImageNet. Recently, [5] trained 3D CNNs on Kinetics dataset and boomed the performance, which also made it possible for us to use a 3D pre-trained model. Thus, we can now use a Kinetics datasets pre-trained model to perform my action recognition.

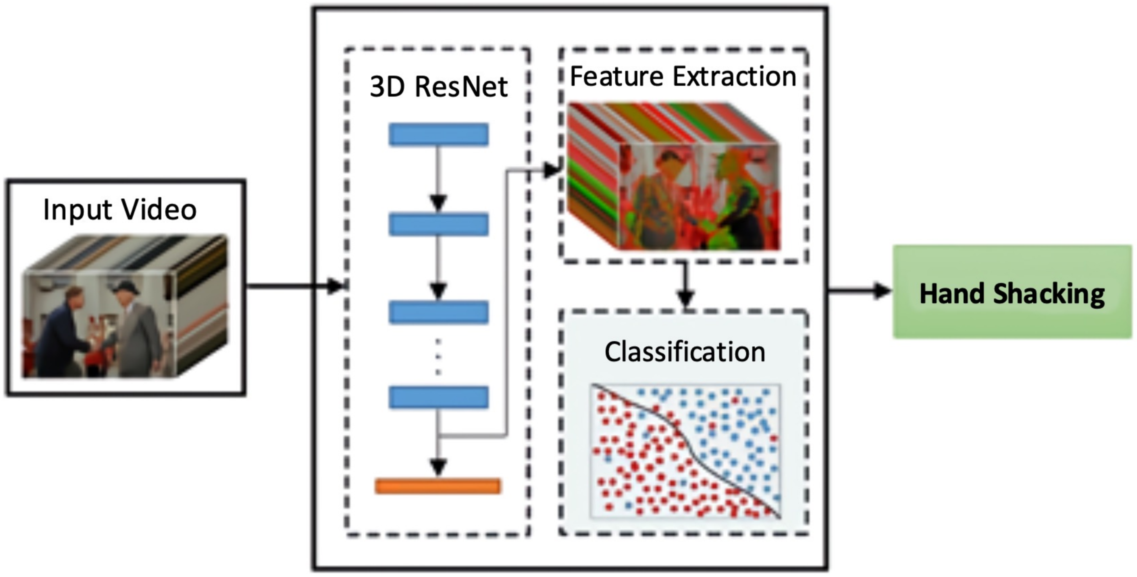

Figure 4.1: Proposed framework including feature extractor and classifier.

In this chapter we propose a BoVW framework. My network contains two parts, a feature extractor and a classifier [Fig. 4.1]. I use a 3D residual networks (ResNet) [15] pre-trained model as my feature extractor and for the classifier, we proposed a Single Layer Feedforward Network with Subnetwork Nodes (SLFN). I tested the ResNet model of different structures from a shallower to a deeper network model using the UCF-101 and HMDB-51 datasets in order to ascertain which structure has the best performance. Additionally, we also tested the feasibility of the pre-trained model. I optimized my SLFN parameters to achieve a better performance. In addition, we evaluated the approach we proposed in terms of accuracy and time consumption. Furthermore, other classifiers, such as SVM, can be used in the method as well. My proposed method could use any type of videos. The proposed framework is summarized in the following section.

4.3 Method

4.3.1 Network structure

The BoVW structure plays a powerful role in image recognition. The general idea of BOVW is to reduce the dimensions of an image or video and encode it into a set of features. Features comprise key points and descriptors. The key points are the ”salient points” in the image, so the key points are the same whether the image is rotated, shrunk, or expanded. A descriptor is a description of the key points. So we can represent each image in terms of the frequency of its features and by virtue of its feature frequency, we can predict the category of another image. In order to explore whether this structure is used in the field of action recognition, we built my network, see Fig. 1 below, and tested the performance.

4.3.2 3D ResNet

For the extractor, we focused on 3D CNNs that have begun to perform better than 2D CNNs on large-scale video datasets. Recently, [13] conducted a series of experiments using different depth 3D ResNet; they also compared the performance between a base model and a pre-trained model. Their experiments showed that the ResNet-101 pre-trained model demonstrated the best performance.

In my study, based on those experiments results provided by [13], we choose the 3D ResNet pre-trained model as my feature extractor. To ensure the accuracy of the results, we reevaluated all experiments. I also tested the performance of 3D ResNet with different depths. The results of my experiment are shown in the following section.

4.3.3 Feature extraction from 3D ResNet models

As mentioned in the previous subsection, the 3D ResNet models have been considered. This model was previously trained on a Kinetics dataset with obtained impressive results. ResNet is known to be a deep CNN model that consists entirely of several convolution layers but only one fully connected layer. Average-pooling layers are employed after convolution layers. As shown in Fig. 4.2 we extracted deep features from the average pool layer.

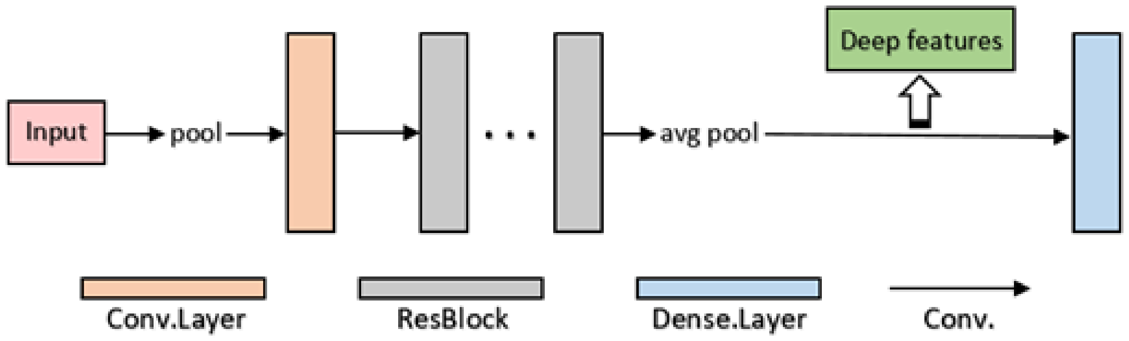

Figure 4.2: Architecture of the 3D ResNet.

4.3.4 Single-layer classifier with sub-network nodes

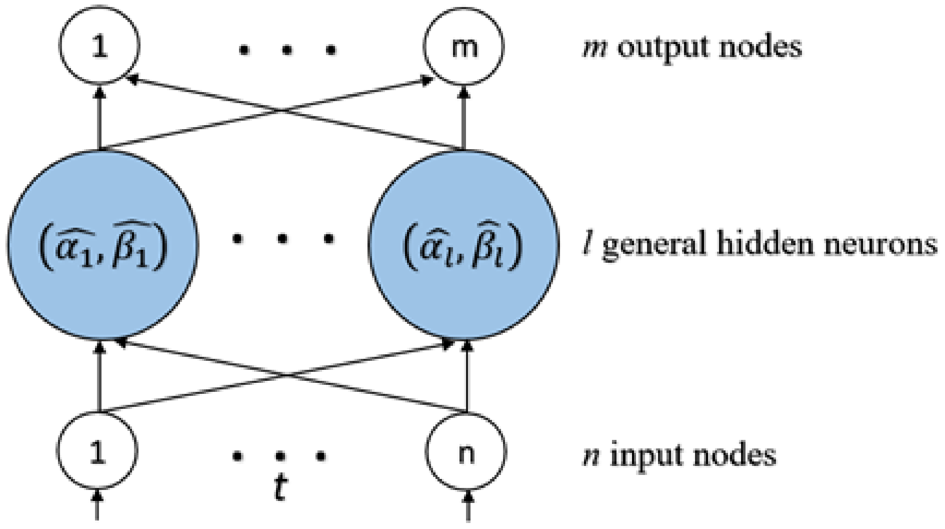
1
m
m output nodes
$(\widehat{\alpha_1}, \widehat{\beta_1})$
$(\widehat{\alpha_l}, \widehat{\beta_l})$
l general hidden neurons
1
n
t
n input nodes

Figure 4.3: My proposed classifier.

In video data processing, time cost and computation cost are often relatively high; so, my proposed new classifier greatly reduces time and computational cost compared to traditional classifiers such as SVM and ELM, and iterative training tends to achieve better results.Fig. 4.3 shows the structure of my classifier.

As we have encoded features data, we will use the proposed SLFN on the encoded dimension data to classify objects with L numbers of sub-networks. Thereafter, we will split the data along with target label data with n size and send it to the network to train it. First, we will use (x_0, t_0) chunk of data for the initial training of the network. Thereafter, e_i will represent the residual network error and $(\hat{\alpha_i}, \hat{\beta})$ will define the input weight and output weight which are going to be updated in every iteration.

$$\alpha(0) = \mathbf{x}_0^{-1} \cdot h^{-1}(e_0) = (\mathbf{x}_0^T \mathbf{x}_0 + (I_{d\times d}/c))^{-1} \mathbf{x}_0^T h^{-1}(e_0) \tag{4.1}$$

where $h^{-1}(\cdot)$ has been used as the inverse function of h$(\cdot)$. I would use m_0 to represent $\mathbf{x}_0^T \mathbf{x}_0 + (I_{d\times d}/c)$. Now we can write Eq. (3.1) in following way:

$$\alpha(0) = m_0 \mathbf{x}_0^T h^{-1}(e_0) \tag{4.2}$$

After the initial training, we will update the input weight in a sequential manner with the next batch of training samples (x_i, t_i), I combine x_0 and x_1 together as well as their corresponding residual error e_0 and e_1, theoretically, we can get the following:

$$\hat{\alpha_1} = \mathbf{m}_1^{-1} \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix}^T \begin{bmatrix} h^{-1}(e_0) \\ h^{-1}(e_1) \end{bmatrix} \tag{4.3}$$

where

$$\begin{cases} \mathbf{m}_1 = I_{d\times d}/c + \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix}^T \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix} \\ \mathbf{m}_1 = I_{d\times d}/c + \mathbf{x}_0^T \mathbf{x}_0 + \mathbf{x}_1^T \mathbf{x}_1 \\ \mathbf{m}_1 = \mathbf{m}_0 + \mathbf{x}_1^T \mathbf{x}_1 \end{cases} \tag{4.4}$$

and

$$\begin{cases} \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix}^T \begin{bmatrix} h^{-1}(e_0) \\ h^{-1}(e_1) \end{bmatrix} & = \mathbf{x}_0^T h^{-1}(e_0) + \mathbf{x}_1^T h^{-1}(e_1) \\ & = \mathbf{m}_0 \mathbf{m}_0^{-1} \mathbf{x}_0^T h^{-1}(e_0) + \mathbf{x}_1^T h^{-1}(e_1) \\ & = \mathbf{m}_0 \alpha_0 + \mathbf{x}_1^T h^{-1}(e_1) \\ & = (\mathbf{m}_1 - \mathbf{x}_1^T \mathbf{x}_1)\alpha_0 + \mathbf{x}_1^T h^{-1}(e_1) \\ & = \mathbf{m}_1 \alpha_0 - \mathbf{x}_1^T \mathbf{x}_1 \alpha_0 + \mathbf{x}_1^T h^{-1}(e_1) \end{cases} \tag{4.5}$$

According to Eq. (3.3), (3.4) and (3.5), we derive

$$\begin{cases} \hat{\alpha_1} & = \mathbf{m}_1^{-1}[\mathbf{m}_1\alpha_0 - \mathbf{x}_1^{-1}\mathbf{x}_1\alpha 0 + \mathbf{x}_1^T h^{-1}(e_1)] \\ & = \alpha_0 - \mathbf{m}_1^{-1}\mathbf{x}_1^T\mathbf{x}_1\alpha_0 + \mathbf{m}_1^{-1}\mathbf{x}_1^T h^{-1}(e_1) \\ & = \alpha_0 + \mathbf{m}_1^{-1}\mathbf{x}_1^T[h^{-1}(e_1) - e_1\alpha_0] \end{cases} \tag{4.6}$$

I can generalize Eq. (3.6) to

$$\hat{\alpha_{i+1}} = \alpha_i + \mathbf{m}_{i+1}^{-1}\mathbf{x}_{i+1}^T[h^{-1}(e_{i+1}) - \mathbf{x}_{i+1}\alpha_i] \tag{4.7}$$

Instead of calculating $(\hat{\alpha_n})$ for each epoch, we can use the previous knowledge and update it by passing new chunk of encoded. Suppose the chunk of data we send for initial training is $\mathbf{x}_{en\_init}$ and remaining data will be considered as $\mathbf{x}_{en\_seq}$. If total numbers of training epochs are $Total\_Epochs$ and batch size for sequential data is $BATCH\_SIZE$. I will train my network in the following manner:

Algorithm 1 OS-Subnetwork training algorithm

```
Result: Update $\alpha_L$ sequentially and calculate corresponding $\beta_L$
Split the dataset for initial and sequential training;
 $epoch \leftarrow 0$;
 For $(\mathbf{x}_{en\_init}, \mathbf{t}_{en\_init})$, we obtain $\alpha_L$and $\beta_L$ for each subnetwork
 while $epoch < Total\_Epochs$ do
     $l \leftarrow 0$;
     while $l < length(x_{en\_seq})$ do
         if $l + BATCH\_SIZE \leq length(x_{en\_seq})$ then
             $\mathbf{x}_{batch} \leftarrow \mathbf{x}_{en\_seq}[l : l + BATCH\_SIZE]$;
             $\mathbf{t}_{batch} \leftarrow \mathbf{t}_{en\_seq}[l : l + BATCH\_SIZE]$;
         else
             $\mathbf{x}_{batch} \leftarrow \mathbf{x}_{en\_seq}[l :]$;
             $\mathbf{t}_{batch} \leftarrow \mathbf{t}_{en\_seq}[l :]$;
         end
         $l \leftarrow l + BATCHSIZE$;
         Update sequentially $\alpha_L$ and calculate $\beta_L$;
     end
     $epoch \leftarrow epoch + 1$;
end
```

The performance of the proposed algorithm depends on both the number of hidden neurons m (encoding dimension) and the pre-defined constant c. A proper way of selecting the optimal value of c is chosen by the trial-and-error method [18]. Now, we will use the Online Sequential-Subnetwork on encoded dimension data to classify objects with L numbers of sub-networks.

4.4 Experiment results

4.4.1 Dataset

The HMDB-51 [21] and UCF-101 [41] datasets are currently the most successful in the field of action recognition. UCF101 has a total of 13320 videos, including 101 action categories. Moreover, video in this dataset have diverse actions, with very different camera movements and often messy background. It is one of the most challenging data sets available. The videos in 101 action categories are divided into 25 groups, where each group can consist of 4-7 videos of an action. The videos from the same group may share some common features, such as similar background, similar viewpoint and so on. The HMDB51 dataset contains 6849 clips divided into 51 action categories, each containing a minimum of 101 clips. Fig. 4.4 shows samples from the video frames of datasets utilized to evaluate my method performance.

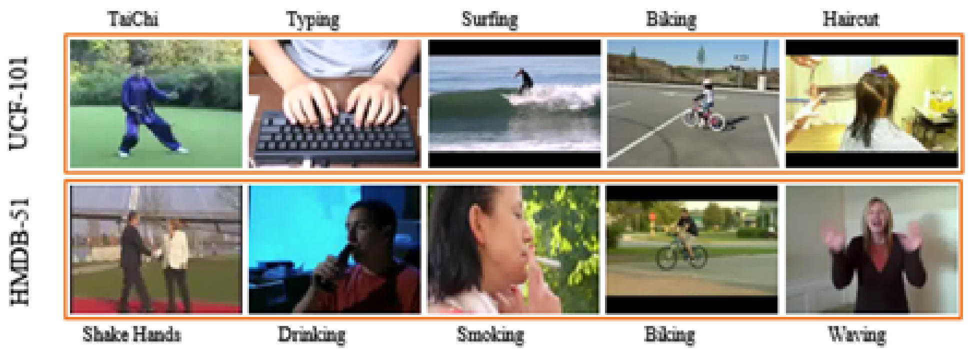

Figure 4.4: The samples of video frames from UCF-101 and HMDB-51 datasets.

In the training and testing process, it is very important to separate the video belonging to the same group. Since videos within a group are all from a single long video, sharing videos from the same group in the training set and testing set can achieve higher performance. So, each of these datasets provide train and test splits files. I evaluated my network performance based on these splits files and calculated average performance.

4.4.2 Environment setting

To test my proposed method, we compare my work with different depth 3D ResNet and classifiers; we also compare my proposed method with other state-of-the-art methods. To find the best performance of my extractor, we compared it with pre-trained 3D ResNet with different depth and learning from scratch. I compared the classifier with Support Vector Machine (SVM), Extreme Learning Machine (ELM), K-nearest neighbors (KNN) and Random Forest (RF) [4]. For SVM and ELM, parameters were set up from $[2^{-10}, 2^{-9}, \ldots, 2^{10}]$ in each experiment. For KNN we set K to 3 and we use auto algorithm. I selected 100 as my RF estimators. For my proposed classifier, the regularization parameter C is selected from $C \in [2^{-4}, \ldots, 2^{8}]$. To test the efficiency of my algorithm, we run my experiments on 2 datasets, which are conducted on a machine with an NVidia GTX-1080Ti GPU.

4.4.3 Extractor evaluation

According to a previous study [13], 3D ResNet trained on UCF-101 and HMDB-51 does not achieve high accuracy whereas a Kinetics pre-trained model works well. In this section, aiming to find the optimal feature extractor, we tried to reproduce the performance in the experiment. In this process, we trained 3D ResNets with different depths by UCF-101 and HMDB-51 dataset from scratch and then we trained Kinetics pre-trained 3D ResNet models as well. Based on previous study [13], we have known that when we devide the video in to 15 frames, 3D ResNet has the best performance. To make the result fairly, we use train and test split file 1; choose batch size of 32, and train 50 epochs. The performances are shown in Tab. 4.1.

As Tab. 4.1 shows Kinetics pre-trained models perform significantly better then learning from scratch. 3D ResNet-101 has the best performance both for learning from scratch and transfer learning. It indicates that the 3D ResNet-101 pre-trained model can learn optimal features more accurately in less time compared to other methods. Therefore, we choose it as my extractor and use it in later experiments.

Model	UCF-101	HMDB-51
Learning from scratch		
3D ResNet-18	35.9	10.1
3D ResNet-34	33.3	12.5
3D ResNet-50	34.5	12.8
3D ResNet-101	38.1	14.1
Transfer learning		
3D ResNet-18	71.4	41.6
3D ResNet-34	76.4	44.3
3D ResNet-50	75.6	46.4
3D ResNet-101	78.9	47.8

Table 4.1: Performances of 3D ResNets

4.4.4 Classifier evaluation

After the experiment above, we choose a 3D ResNet-101 pre-trained model as my extractor. In this section we evaluated my proposed classifier's performance. A performance comparison has been evaluated among SVM, ELM, KNN, RF and my proposed algorithm. For the accuracy and fairness of the experiment, we first trained an extractor and extracted deep features; later, the classifier recognized the actions. Further, we carried out this experiment in Tab. 4.4 to compare my single-layer network with those learning algorithms. Here, it can be seen that the accuracy of my method is clearly higher compared that of other classifiers.

Model	UCF-101	HMDB-51
3D ResNet-101+SVM	83.3	50.8
3D ResNet-101+ELM	84.7	53.7
3D ResNet-101+KNN	82.3	48.6
3D ResNet-101+Rf	43	39.1
3D ResNet-101+**Mys**	85.3	54.5

Table 4.2: Performances of different classifiers

I also evaluated the time consumption, and the comparison results of time consumption are shown in Tab. 3, which indicates that my classifier has seen a significant improvement in testing speed compared to SVM. Further, the ELM testing speed is similar. However, my classifier supports iterative training, and the corresponding batch size can be set according to the situation, which is especially important in video processing and in especially large video datasets.

Model	UCF-101	HMDB-51
SVM	486	116
ELM	10	3.7
KNN	567	85
Rf	18.6	4.8
Mys	2.2	0.96

Table 4.3: Time consumptions of different classifier (s)

4.4.5 Framework evaluation

The above experiments show that a Kinetics dataset can be used to train my network. Comparisons with other state-of-the-art architectures are shown in Tab. 4.4. As can be seen from Tab. 4.4, my method achieved higher accuracies compared with 3D Resnet-101, 3D ResNext-101 [13], C3D [44], P3D [30], and two-stream I3D [5]. I can also observe that two-stream I3D, which is pre-trained by the Kinetics dataset, achieves the best accuracy. In addition, we believe that combining the two-stream architecture with my framework can further improve the accuracy of two-stream I3D.

Model	UCF-101	HMDB-51
3D ResNet-101	78.9	10.1
3D ResNext-101[13]	81.4	50.3
C3D[44]	78.1	-
P3D[30]	80.2	-
Two-stream I3D[5]	**92.5**	**63.7**
Mys	85.3	54.5

Table 4.4: Performances of different classifiers

4.5 Conclusion

In this chapter, we tested various CNNs architectures of spatio-temporal three-dimensional convolution kernels on the current video dataset. According to these experimental results, the following conclusions can be drawn: (1) The BoVW structure is efficient on video processing. (2) It is effective for 3D CNNs to pre-train on the Kinetics dataset, which has sufficient data to optimize the 3D CNN network. (3) Instead of using randomized input weights, we can approach a classifier where weights would be configured by calculation and reach to the steepest descent in iterative manner without configuring the learning rate.

Chapter 5

Combination of Multi-stream Features for EEG-Based Signal Emotion Recognition

5.1 Overview 43
5.2 Introduction 43
5.3 Method 45
5.3.1 First General Layer: Feature Extraction 45
5.3.2 Second general layer: Early fusion 47
5.3.3 Third General Layer: Classifier with Sub-network Nodes . . . 48
5.3.4 Generate Inputs to CNN Auto-encoder 48
5.4 Experiments 52
5.4.1 Dataset 52
5.4.2 Environment Setting 53
5.4.3 SEED Dataset 53
5.4.4 SEED-V dataset 56
5.5 Conclusion 59

5.1 Overview

In the previous chapter, we improved the accuracy of motion recognition through 3D feature extraction, but my method was not state-of-the-art. In the previous chapter, we cited a novel feature extraction method. That is, two-stream feature extraction and fusion, achieved the state-of-the-art results in action recognition. I believe that EEG feature extraction is an important part of emotion recognition and could offer a significant computational advantage over recognition accuracy. However, based on two-stream feature extraction method, EEG features can be further encoded or extracted to have better recognition accuracy and generalization performance. Thus, this chapter proposes a multi-stream hierarchical network framework that learning behaviors of features combined from multi networks. Features of EEG signal were extracted from temporal, spatial and frequency domain respectively, and the impact of fusion of different features on the recognition accuracy was tested and analyzed. I evaluated my framework on the SEED dataset and SEED-V dataset and compared with other methods, we achieved state-of-the-art results.

5.2 Introduction

In the past few years, many good feature descriptors have appeared for target recognition. Many methods divide the input data into dense patches arranged regularly, and then extract the features of these patches. The characteristics of these patches are then combined in some way as the characteristics of this input data. In a nutshell, a large part of these systems is a feature extraction process: the input passes through a filter bank filter bank (generally an edge detector based on directionality), and then passes through a non-linear operator non-linear operation (quantization, winner-take-all, sparsification, normalization, and point-wise saturation), and then use a pooling operation (pass the value of the real space or feature space neighborhood through a max, average, or histogramming operator) to The peacekeeping gets a certain invariance. For example, the SIFT feature, which we are familiar with, first passes the directional edge detector for each small patch, and then uses the winner-take-all operator to obtain the most significant direction. Finally, the histogram of the local direction is counted on the larger patch, and pooled into a sparse vector.

For a layer of feature extraction system, that is, after extracting the above feature, such as SIFT, HOG, etc., and then directly connecting a supervised learning

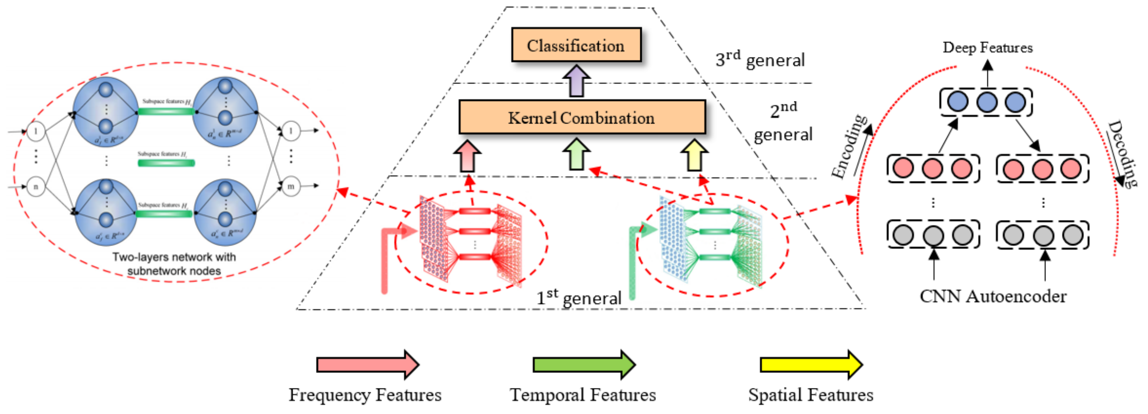

Figure 5.1: Proposed framework, 1) 1^{st} general layer: feature extraction. 2) 2^{nd} general layer: early fusion. 3) 3^{rd} general layer: classifier with subnetwork nodes.

classifier, it constitutes a target recognition system. Some models use two or more levels of feature extractors, and then a supervised learning classifier to form a more complex target recognition system. The essential differences between these systems are: there are one or more feature extraction layers, non-linear operators used after the filter bank, the filter bank is obtained (manual selection, unsupervised learning or supervised learning), and the selection of the top classifier (A linear classifier is a more complex classifier.)

Generally, the choice of filter bank is Gabor wavelet, and some people choose some simple directional detection filter bank, that is, gradient operators, such as SIFT and HOG. There are also those filter banks that are learned directly from training data through unsupervised learning methods. When training on natural images, the learned filters are similar to Gabor edge detection. A benefit of the feature learning method is that it can learn features hierarchically. Because we have a certain prior knowledge, for secondary features, humans do not have similar prior knowledge. So it is more difficult to manually design a better secondary feature extractor. So, the second-level or multi-level features must let the system learn by itself. There are a lot of methods now, supervised, unsupervised, or a combination of the two.

In Chapter 3, we improved the accuracy of emotion recognition by fusing spatial features with frequency-domain features. [2] proposed using the Long short-term memory (LSTM) [17] network for emotion recognition based on EEG signals, and achieved good results, proving that the EEG signals are continuous in the time domain. Their method also proved that EEG EEG signals are time-domain. Based on [10] method, time-domain features can be extracted for any datasets with time-domain features to increase recognition accuracy.

I extract DE features from the EEG 1-dimensional raw data (see 3.3.1 for the extraction method). I call the DE features a first-level feature. Through the DE feature extraction algorithm, we can know that the DE features are also continuous in the time domain, which means that DE features can extract secondary features in the time domain.

Motivated by these evidences, this chapter proposes an multi-stream method that fusing temporal, spatial and frequency features for emotion recognition [see Fig. 5.1]. In particular, this chapter makes the following contributions.

1. I propose a multi-stream hierarchical network framework learning behaviors of features combined from multi networks. Each stream can extract temporal, spatial and frequency features respectively, which significantly improve accuracy. I adopted an early fusion method that is better than later fusion by fusing different features into one super vector.

2. My proposed method is based on the extraction of second-level features. The experimental results prove that the second-level features have better ability to describe the input data than the first-level features.

3. Superb generalization performance. My method is evaluated on two datasets (SEED and SEED-V), we tested my framework by using original EEG signals, DE features and PSD features and compared with several state-of-the-art methods and we got the best performance.

5.3 Method

5.3.1 First General Layer: Feature Extraction

Two-layers Auto-encoder Network

A two-layer auto-encoder has been introduced [50] where only the encoding layer weight has been generated randomly, based on which the decoding layer weight has been calculated. The auto-encoder aims is to minimize the reconstruction loss, which is the squared error between the input X and the neural network output $\hat{Y}$.

I briefly describe the Auto-encoder algorithm in the following steps.

Step-1: Given M arbitrary distinct training samples $(\mathbf{x}_k, \mathbf{y}_k)_{k=1}^{M}, \mathbf{x}_k \in \mathbf{R}^n$ are sampled from a continuous system. Randomly initialize the entrance layer sub-network node:

$$\mathbf{H}_f^c = g(\hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c, \mathbf{x}), (\hat{\mathbf{a}}_f^c)^T \cdot \hat{\mathbf{a}}_f^c = 1, (\hat{\mathbf{b}}_f^c)^T \cdot \hat{\mathbf{a}}_f^c = 1 \tag{5.1}$$

where $\hat{\mathbf{a}}_f \in \mathbf{R}^{d\times n}, \hat{\mathbf{b}}_f \in \mathbf{R}$ is the orthogonal random weight and bias of the entrance mapping layer. $\mathbf{H}_f^c$ is the c-th subspace features. c represents sub-network node index and initial index $c = 1$.

Step-2: Given an invertible activation function g, obtain the sub-network node of the exit feature layer $(\hat{\mathbf{a}}_h^c, \hat{\mathbf{b}}_h^c)$ by

$$\begin{aligned} &\hat{\mathbf{a}}_h^c = g^{-1}(u_n(\mathbf{y})) \cdot (\mathbf{H}_c^c)^{-1}, \hat{\mathbf{a}}_h^c \in \mathbf{R}^{d\times m}, \\ &\hat{\mathbf{b}}_h^c = \sqrt{mse(\hat{\mathbf{a}}_h \cdot \mathbf{H}_f^c - g^{-1}(u_n(\mathbf{y})))}, \hat{\mathbf{b}}_h^c \in \mathbf{R} \end{aligned} \tag{5.2}$$

where $\mathbf{H}^{-1} = \mathbf{H}^T(C1/I + +\mathbf{H}\mathbf{H}^T)^{-1}; C1 > 0$ is a regularization value; u_n is a normalized function $u_n(\mathbf{y}) : \mathbf{R} \rightarrow (0, 1]; g^{-1}$ and u_n^{-1} represent their reverse function.

Step-3: Update the output error $\mathbf{e}_c$ as

$$\mathbf{e}_c = \mathbf{y} - u_n^{-1} g(\mathbf{H}_f^c, \hat{\mathbf{a}}_h^c, \hat{\mathbf{b}}_h^c) \tag{5.3}$$

I can get error feedback data $\mathbf{P}_c = g^{-1}(u_n(\mathbf{e}_c)) \cdot (\hat{\mathbf{a}}_h^c)^{-1}$.

Step-4: Update the sub-network node $\hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c$ in the entrance layer

$$\begin{aligned} &\hat{\mathbf{a}}_f^c = g^{-1}(u_j(\mathbf{P}_{c-1})) \cdot \mathbf{x}^{-1}, \hat{\mathbf{a}}_f^c \in \mathbf{R}^{n\times d}, \\ &\hat{\mathbf{b}}_f^c = \sqrt{mse(\hat{\mathbf{a}}_f^c \cdot \mathbf{x} - \mathbf{P}_{c-1})}, \hat{\mathbf{b}}_f^c \in \mathbf{R} \end{aligned} \tag{5.4}$$

Step-5: obtain the c-th subspace feature data

$$\mathbf{H}_f^c = g(\mathbf{x}, \hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c) \tag{5.5}$$

Step-6: Set $c = c + 1$, add a new sub-network node $\hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c$ in the feature mapping layer with orthogonal random initialization.

Step-7: Repeat steps 2 to 6 $L - 1$ times, then obtain the L subspace features $[\mathbf{H}_f^1, ..., \mathbf{H}_f^L]$.

CNN Auto-encoder

In Chapter 3 [see 3.3], we proposed a CNN-based auto-encoder network structure, which achieved good results. In this chapter, we continue to use the network to extract spatial and time domain features.

5.3.2 Second general layer: Early fusion

Since we extract features (temporal, spatial, frequency) through multiple streams, we need to find a way to fuse these features. [35] and [52] show that If there is correction information between different features, then the simple stitching of features into a super vector, then early fusion can promote later fusion. For example, we use two different networks to extract two different features. I define those features extracted from the first network as $\mathbf{H}^1 = \left\{\mathbf{H}^1_1, \mathbf{H}^1_2, ..., \mathbf{H}^1_c\right\}$, and features extracted from the second network as $\mathbf{H}^2 = \left\{\mathbf{H}^2_1, \mathbf{H}^2_2, ..., \mathbf{H}^2_c\right\}$. After early fusion, we can generate composite features.

$$\mathbf{H}^{1\bigoplus 2} = [\mathbf{H}^1_1, \mathbf{H}^1_2, ..., \mathbf{H}^1_c, \mathbf{H}^2_1, \mathbf{H}^2_2, ..., \mathbf{H}^2_c]^T \tag{5.6}$$

Given that several features $\mathbf{H}^1, ..., \mathbf{H}^c$, K represent combinatorial operators, combinational features can be expressed as

$$\begin{aligned} \mathbf{H}^{1\bigoplus 2} &= K(\mathbf{H}^1, \mathbf{H}^2), \\ \mathbf{H}^{1\bigoplus 2\bigoplus 3} &= K(K(\mathbf{H}^1, \mathbf{H}^2), \mathbf{H}^3), \\ &\vdots \\ \mathbf{H}^{1\bigoplus 2\cdots\bigoplus c} &= K(\ldots K(K(\mathbf{H}^1, \mathbf{H}^2), \mathbf{H}^3)\ldots). \end{aligned} \tag{5.7}$$

In the field of biology, the brain's mixed neurons play an important role in the realization and coding of brain functions. The brain uses subspace features generated by neurons to eliminate related factors. At the same time, the brain reorganizes subspace features and fuses them into complex and stable behaviors. Therefore, the method we use conforms to the biological structure. At the fusion layer, we use the training samples in the final classifier through the early fusion method.

For my framework, any type of data can be extracted and combined directly. My approach has multiple features that can be summarized in the following sections.

5.3.3 Third General Layer: Classifier with Sub-network Nodes

I adopted my previous classifier [49] as my final classifier. Given $\mathbf{M}$ distinct feature samples combined from combination operator $(\mathbf{H},\mathbf{t})$. $u(x) : \mathbf{R} \rightarrow (0,1]$ is a normalized function; g is a sigmoid or sine activation function, and then for any continuous outputs t, we have $lim_{n\rightarrow+\infty}\|\mathbf{t} - ((g(\mathbf{a}_p^1, \mathbf{b}_p^1, \mathbf{H})) \cdot \boldsymbol{\beta}_p^1 + \cdots + (g(\mathbf{a}_p^c, \mathbf{b}_p^c, \mathbf{H})) \cdot \boldsymbol{\beta}_p^c)\| = 0$ holds with probability one if

$$\begin{aligned} \boldsymbol{a}_p^c &= g^{-1}((e_{c-1})) \cdot \mathbf{H}^T \left(\frac{C}{\mathbf{I}} + \mathbf{H}\mathbf{H}^T\right)^{-1}, \\ b_p^c &= \text{sum}(\mathbf{a}_p^c \cdot \mathbf{H} - h^{-1}((\mathbf{e}_{c-1})))/N, b_p^c \in \mathbf{R}. \end{aligned} \tag{5.8}$$

$$\begin{aligned} g^{-1}(\cdot) &\begin{cases} \arcsin(\cdot) & \text{if } g(\cdot) = \sin(\cdot), \\ -\log\left(\frac{1}{(\cdot)} - 1\right) & \text{if } g(\cdot) = 1/(1+e^{-(\cdot)}), \end{cases} \\ \mathbf{e}_c &= \mathbf{t} - u_n^{-1} g(\mathbf{H}, \mathbf{a}_p^c, \mathbf{b}_p^c), \\ \boldsymbol{\beta}_p^c &= \frac{\langle \mathbf{e}_{c-1}, u^{-1}(g(\mathbf{a}_p^c \cdot \mathbf{H} + b_p^c))\rangle}{\|u^{-1}(g(\mathbf{a}_p^c \cdot \mathbf{H} + b_p^c))\|^2}. \end{aligned} \tag{5.9}$$

where $[\cdot]^{-1}$ represents its inverse function.

5.3.4 Generate Inputs to CNN Auto-encoder

Spatial Inputs

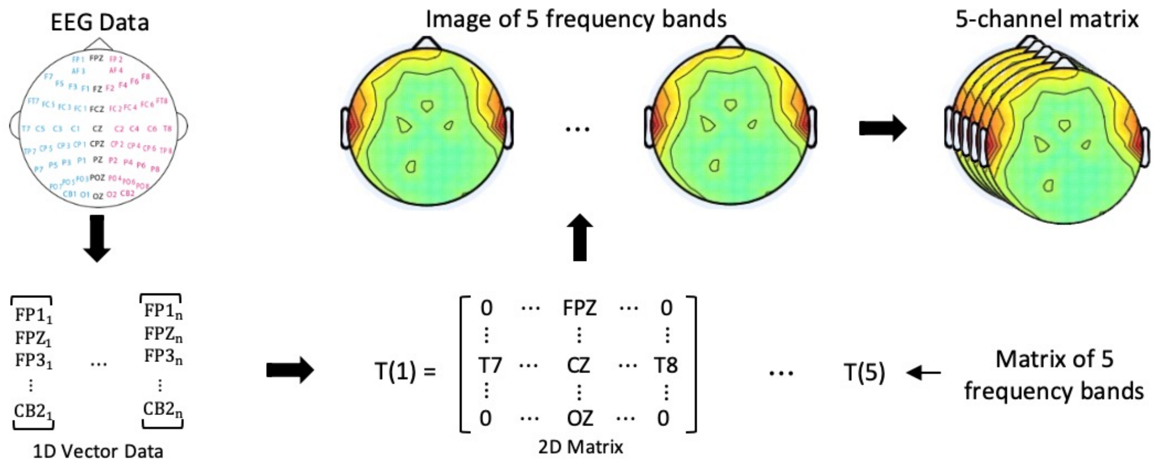

Figure 5.2: Converting EEG-based signals to matrix.

EEG signals are collected at different spatial locations in the cerebral cortex and are composed of multiple time series. Therefore, EEG signals have additional spatial information.

The one-dimensional data obtained by analyzing this measurement method is the standard method in EEG data analysis. However, this approach obviously ignores the relationship between space and frequency. Therefore, we convert the measured values into a two-dimensional matrix to preserve the spatial structure and use multiple channels to represent the spectrum of bands with different frequencies.

Both DE and power spectral density (PSD) features contains the values of 62 electrodes in five frequency bands. The detailed order of the channels is included in the dataset. The EEG cap according to the international 10-20 system for 62 channels.

Filters in convouational neural networks (CNNs) can learn patterns of adjacent values in the connectivity matrix, which enables CNN to learn spatial features of a two-dimensional matrix. In order to generate the input matrix data of CNN, we need to convert the EEG feature data as two-dimensional matrices. I converted the EEG data into a 9×9 matrix, for the boundaries of the matrix and positions of the null electrodes are padding with zeros. The matrix can be defined as

$$T(n) = \begin{bmatrix} 0 & 0 & 0 & FP1 & FPZ & FP2 & 0 & 0 & 0 \\ 0 & 0 & 0 & AF3 & 0 & AF4 & 0 & 0 & 0 \\ F7 & F5 & F3 & F1 & FZ & F2 & F4 & F6 & F8 \\ FT7 & FC5 & FC3 & FC1 & FCZ & FC2 & FC4 & FC6 & FT8 \\ T7 & C5 & C3 & C1 & CZ & C2 & C4 & C6 & T8 \\ TP7 & CP5 & CP3 & CP1 & CPZ & CP2 & CP4 & CP6 & TP8 \\ P7 & P5 & P3 & P1 & PZ & P2 & P4 & P6 & P8 \\ 0 & PO7 & PO5 & PO3 & POZ & PO4 & PO6 & PO8 & 0 \\ 0 & 0 & CB1 & O1 & OZ & O2 & CB2 & 0 & 0 \end{bmatrix} \tag{5.10}$$

Where n represents frequency bands. So the EEG data meshes D is created as follows:

$$D(n) = [T_1, T_2, ..., T_5] \tag{5.11}$$

Temporal Inputs

In this section, we describe a network use temporal recognition stream as input. The input of the model is generated by stacking the displacement of several consecutive frames of data, which clearly describes the motion trajectory between signal points. Therefore, the network itself does not need the ability to learn motion features, so, we use the same structure as the network that extracts spatial features.

I use $\mathbf{d}_f$ to represent the displacement vector of signal strength from one electrode at frame f, which moves the point to the corresponding point in the next frame $f+1$. Suppose we have a total of M electrode signals for the EEG signal, to represent the motion across a series of frames, we stack the stream d_f^M of N consecutive frames to form a total of $M \times N$ input image. Therefore the input is well suited to recognition using a convolutional network. So, the input volume $\mathbf{I}_f \in \mathbf{R}^{M \times N}$ for an arbitrary frame f can be described as follows:

$$\mathbf{I}_f^m(k) = d_{f+k-1}^m, \quad m = [1 : M], k = [1 : N] \tag{5.12}$$

Furthermore, different structures auto-encoder can be used in my method, other classifiers, such as SVM or ELM, can also be used in the method as well. The proposed algorithm could be summarized in the following Algorithm 2.

Algorithm 2 The proposed method

```
Given a large training dataset $(\mathbf{x}_k, \mathbf{y}_k)_{k=1}^{M}$, $\mathbf{x}_k \in \mathbf{R}^n$, an invertible activation function g, number of hidden nodes in each sub-network node d (d equals number of targeted dimensionality of the subspace features), regularization coefficient C, and the number of sub-network nodes L:
**First general layer: Subspace feature extraction**:
Step 1: Convert EEG-based signal to images and temporal stream by equation (5.10-5.12)
Step 2: Extract frequency features. Set $c = 1$, randomly generate the sub-network node for entrance feature layer by equation (5.1).
**while** $c < L$ **do**
    Calculate the sub-network node for exit feature layer by equation (5.2);
    Calculate the output error and error feedback data by equation (5.3);
    Update the sub-network node $\hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c$ in the entrance layer by equation (5.4);
    Obtain the $c$-th subspace feature data by equation (5.5);
    Set $c = c + 1$, add a new sub-network node $\hat{\mathbf{a}}_f^c, \hat{\mathbf{b}}_f^c$ in the feature mapping layer with orthogonal random initialization (equation (5.1));
    Repeat $L - 1$ times, obtain the $L$ subspace features $\mathbf{H}_1$;
**end**
Step 3: Train CNN auto-encoder by EEG images and extract spacial features $\mathbf{H}_2$.
Step 4: Train CNN auto-encoder by EEG temporal stream and extract temporal features $\mathbf{H}_3$.
**Second general layer Feature combination**
Obtain combined features $\mathbf{H}$ as:

    $$\mathbf{H} = \mathbf{H}^{1 \bigoplus 2 \bigoplus 3} \tag{5.13}$$

**Third general layer: Pattern learning:**
Given combined feature $H$, set $c = 1, e_1 = t$.
**while** $c < L$ **do**
    Step 1: Calculate the $c$th sub-network node $(a_p^c, b_p^c)$, and output weights $\beta_p^c$ as:

    $$\begin{aligned} \boldsymbol{a}_p^c &= g^{-1}((e_{c-1})) \cdot \mathbf{H}^T \left( \frac{C}{\mathbf{I}} + \mathbf{H}\mathbf{H}^T \right)^{-1}, a_p^c \in \mathbf{R}^{n \times m} \\ b_p^c &= \text{sum}(\mathbf{a}_p^c \cdot \mathbf{H} - h^{-1}((\mathbf{e}_{c-1})))/N, b_p^c \in \mathbf{R} \\ \boldsymbol{\beta}_p^c &= \frac{\langle \mathbf{e}_{c-1}, u^{-1}(g(\mathbf{a}_p^c \cdot \mathbf{H} + b_p^c)) \rangle}{\| u^{-1}(g(\mathbf{a}_p^c \cdot \mathbf{H} + b_p^c)) \|^2} \end{aligned} \tag{5.14}$$

    Step 2: Calculate $\mathbf{e}_c = \mathbf{e}_{c-1} - \beta \cdot g(\mathbf{a}_p^c, b_p^c, \beta_p^c)$
**end**
```

5.4 Experiments

5.4.1 Dataset

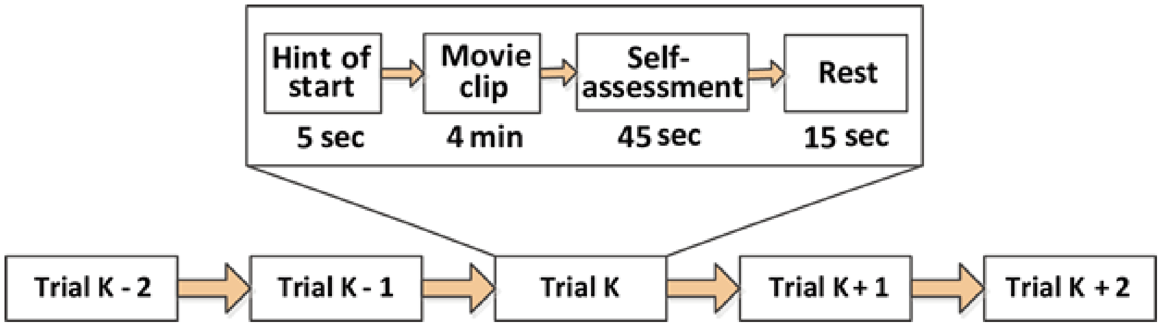

Figure 5.3: Experiment protocol.

Previous studies [9, 55] have shown that raising human emotions through movie clips is reliable. In this chapter, we adopted the SEED [53] and SEED-V [23] dataset, one of the largest databases, which has been popularly used for EEG-based emotion recognition. Video clips with audio were used to elicit specific emotions of the subjects' emotions. Each film clip is about fmy minutes long, and carefully selected important clips enable it to create coherent emotions, which can well trigger the corresponding human emotions. There are 15 clips. Each clip has five seconds of prompts before and 45 seconds of feedback after each clip [see Fig. 5.3]. All movie clips are sorted according to different emotions to ensure that the same emotional movie clips are displayed discontinuously. To test the stability of EEG signals for sentiment analysis over time and the performance of cross-session emotion recognition, All experimental participants were required to conduct 3 trials, each trial being more than 3 days. To ensure the accuracy of emotional records, each subject was asked to complete an Eysenck Personality Questionnaire (EPQ) before the start of each trial. Extroverts who proved stable were selected as subjects. Therefore, subjects who reported themselves as normal participated in the experiment. According to the international 10-20 system, the EEG NeuroScan system was used to record EEG signals at a sampling rate of 1000 Hz. There are 62 active AgCl electrode channels for recording EEG signals. The impedance of each channel in the cap was controlled to less than 5 KΩ.

5.4.2 Environment Setting

In this section, we systematically compare the performance of nine methods for emotion recognition: 1) SVM; 2) Extreme Learning Machine (ELM); 3) Graph regularized Extreme Learning Machine (GELM) [54]; 4) Hierarchical ELM (H-ELM) [42]; 5) DBNs [53]; 6) Bimodal Deep AutoEncoder (BDAE) [22]; 7) HNSN [51]; and 8) Logistic regression (LR); 9) the proposed method. These methods use the above two features (DE and PSD) as input. In order to be consistent with the previous experiment, we used the same environment configuration, as shown in the Table 3.2

5.4.3 SEED Dataset

SEED dataset has three emotion status which are positive, negative and neutral. A total 14 subjects' (7 males and 7 females) data in three time periods or sessions are provided in the dataset.

Subject Dependent Test

Methods	DE
Frequency Stream	91.67
Spatial Stream	88.9
Temporal Stream	76.59
Multi-Stream	**94.84**

Table 5.1: Mean Accuracy of Each Stream and Multi-strem Fusion Approaches

Subject dependence means use a person's emotional responses, stimulated from different film clips to predict this person's emotions. In this experiment, the training sample contains 15 sessions. The training and testing data are from different sessions of the same subject. For cross validation, we randomly split train/test splits (nine sessions as training data, while the rest six sessions as testing data) for three times. For consistency with other methods, we used only the first and last sessions in the SEED dataset. Table 5.1 shows the performance of each stream and my method. For emotion recognition using DE feature we obtain an average accuracy of 94.84%, which is nearly 3% higher than [51] method. I also analyze the confusion matrices of each stream to investigate the complementary characteristics. Fig. 5.4 present the confusion matrices of each stream and the fusion approaches.

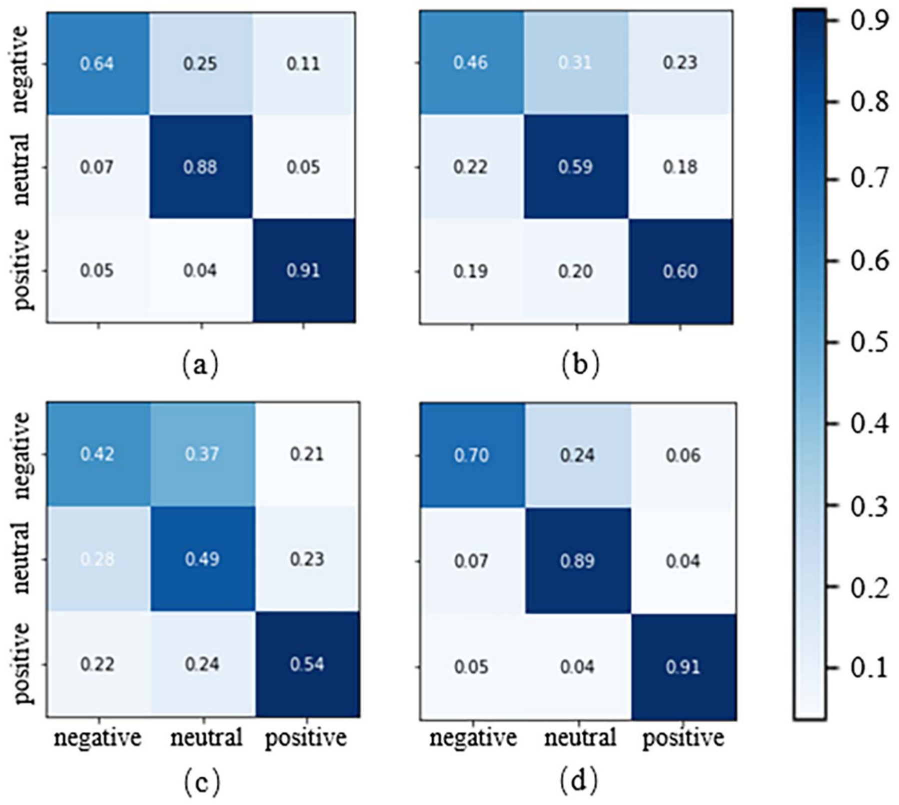

Figure 5.4: The confusion matrices of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream.

For the performance of multi-stream fusion approaches shown in Fig. 5.4, these multi fusion methods can significantly improve the classification performance in negative emotions which improves the accuracies of 6%, compared with a single stream. However, the fusion of these multi-stream has no advantage over a single stream in categorizing neutral and positive emotions.

I compared the accuracy of my proposed methods with SVM, ELM, H-ELM, DBNs [53], LR, Adaptive Subspace Feature Matching (ASFM) [6], Dynamical Convolutional Neural Networks (DGCNN) [39], BDAE [22], GELM [54] and HNSN [51] to show the advantages of my proposed method. Table 5.2 and Fig. 5.5 show the results of comparative experiments. From the table, we can see that the accuracy of my proposed method is significantly higher than that of other methods. In addition, my results are consistent with previous studies [9] and [53]. In EEG-based emotional recognition, DE features have the best performance.

Methods	DE	PSD
SVM	89.99	59.60
ELM	82.92	60.80
LR	82.70	-
DBNs [53]	86.08	61.90
ASFM [6]	83.51	-
DGCNN [39]	90.40	81.73
BDAE [22]	91.01	85.10
GELM [54]	91.07	72.75
HNSN [51]	91.67	73.81
OURS	**94.84**	**83.34**

Table 5.2: Mean Accuracy of Subject Dependent Test

Cross Session Test

Cross session refers to the prediction of a person's emotions through the corresponding emotions of the same person at different times. This tests the stability of emotion recognition model in time domain. As we know that we have total 14 subjects' data in three time periods or sessions, there is a week or longer interval between each session. Compared to other datasets and emotion recognition models, my method is novel in assessing whether the performance of my emotion recognition model is stable over time by performing cross-session testing. For the dataset, each subject has three

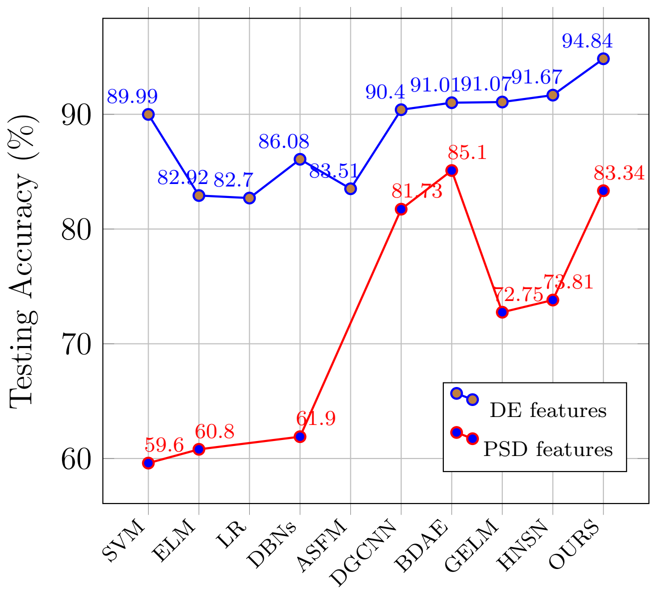

Figure 5.5: Comparison experiment results of Subject Dependent Test.

sessions, conducted on different dates. For cross validation test, we randomly split train/test splits (two time periods data as training data, while the rest one time period data as testing data) for three times. For all the other methods, the way of parameter selection is the same as mentioned in the previous session.

I compared the accuracy of my proposed methods with ELM, H-ELM, SVM, DBN, ASFM, DGCNN, BDAE, GELM, HNSN to show the advantages of my proposed method. Table 5.5 show the performance evaluation of the proposed method and other methods. As can be seen from Table 5.5 and Fig. 5.6, the profit of the proposed test accuracy method is obvious.

The results suggest that the relationship between changes in emotional state and EEG signals is stable for a person over time. The stability of EEG signal decreases with time. The stability of EEG signals in time domain needs to be studied in the future.

5.4.4 SEED-V dataset

In order to show my framework generalization performance and test how my network performs on datasets with more emotional categories, we adopt SEED-V dataset which has five emotion status, disgust, fear, sad, neutral and happy. In the experiment, carefully selected film clips are used as the stimuli, which have been explored

Methods	DE	PSD
SVM	60.95	48.10
ELM	77.62	62.86
H-ELM	80.67	59.05
DBNs [53]	76.57	62.98
ASFM [6]	84.64	-
DGCNN [39]	79.95	64.27
BDAE [22]	66.08	66.23
GELM [54]	79.28	-
HNSN [51]	80.84	61.43
OURS	**88.45**	**68.21**

Table 5.3: Mean Accuracy of Cross Session Test

Methods	DE
Frequency Stream	61.50
Spatial Stream	64.29
Temporal Stream	66.27
Multi-Stream	**71.48**

Table 5.4: Mean Accuracy of Each Stream and Multi-strem Fusion Approaches

to have reliability in eliciting emotions [34]. [23] first proposed and used the SEED-V dataset, and obtained a recognition accuracy of 69.5%. A total of 45 video clips with highly emotional contents are used and edited into 3 segments, each of which consists 15 clips (3 for per emotions). For each segment, 15 clips are placed at random, but for the convenience of subsequent 3-fold cross-validation, the 5 clips in each fold are guaranteed to have different emotion labels. Sixteen healthy subjects (6 males and 10 females) participant in the experiments and each subject is required to perform the experiments for three sessions, at an interval of one week or longer. EEG signal is collected simultaneously when the subjects are watching the film clips.

Experiment results have shown that DE features have the best performance, so we only DE features in the latter experiments. I use the same way to generate three train/test splits as mentioned above. I first compared each stream features approach accuracy [see Tab. 5.4]. As we can see fusing multi stream features achieves the best performance.

To further analysis the effect of multi-stream features on my final accuracy, I analyze the confusion matrices of each stream to investigate the complementary char-

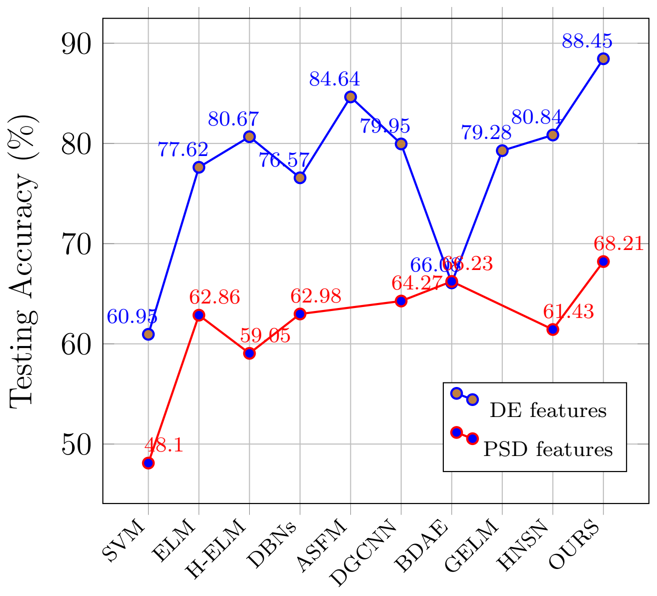

Figure 5.6: Comparison experiment results of Cross Session Test.

acteristics. Fig. 5.7 present the confusion matrices of each stream and the fusion approaches. It can be seen from the Fig. 5.7 that the features of different channels do not complement emotion recognition, but enhance the recognition accuracy of each emotion category.

To see benefits of deep features, extracted by my proposed method, we run at-distributed stochastic neighboring embedding algorithm (t-SNE) to find two-dimensional embeds of high-dimensionalfeature space and plot them as colored points, according tosemantic categories in a particular hierarchy, as shown in Fig. 5.8. Data from each emotion can be gathered in the potential space, and the generated data is close to the corresponding actual data, meaning that the generated data reflects enough realistic information. The generated data complements the training manifold, bringing a better margin for the classifier.

Methods	Subject Dependent Test	Cross Session Test
ELM	33.69	31.28
SVM	57.52	42.59
BDAE [?]	69.50	-
HNSN[51]	61.50	33.3
OURS	**71.48**	**44.29**

Table 5.5: Mean Accuracy of Cross Session Test

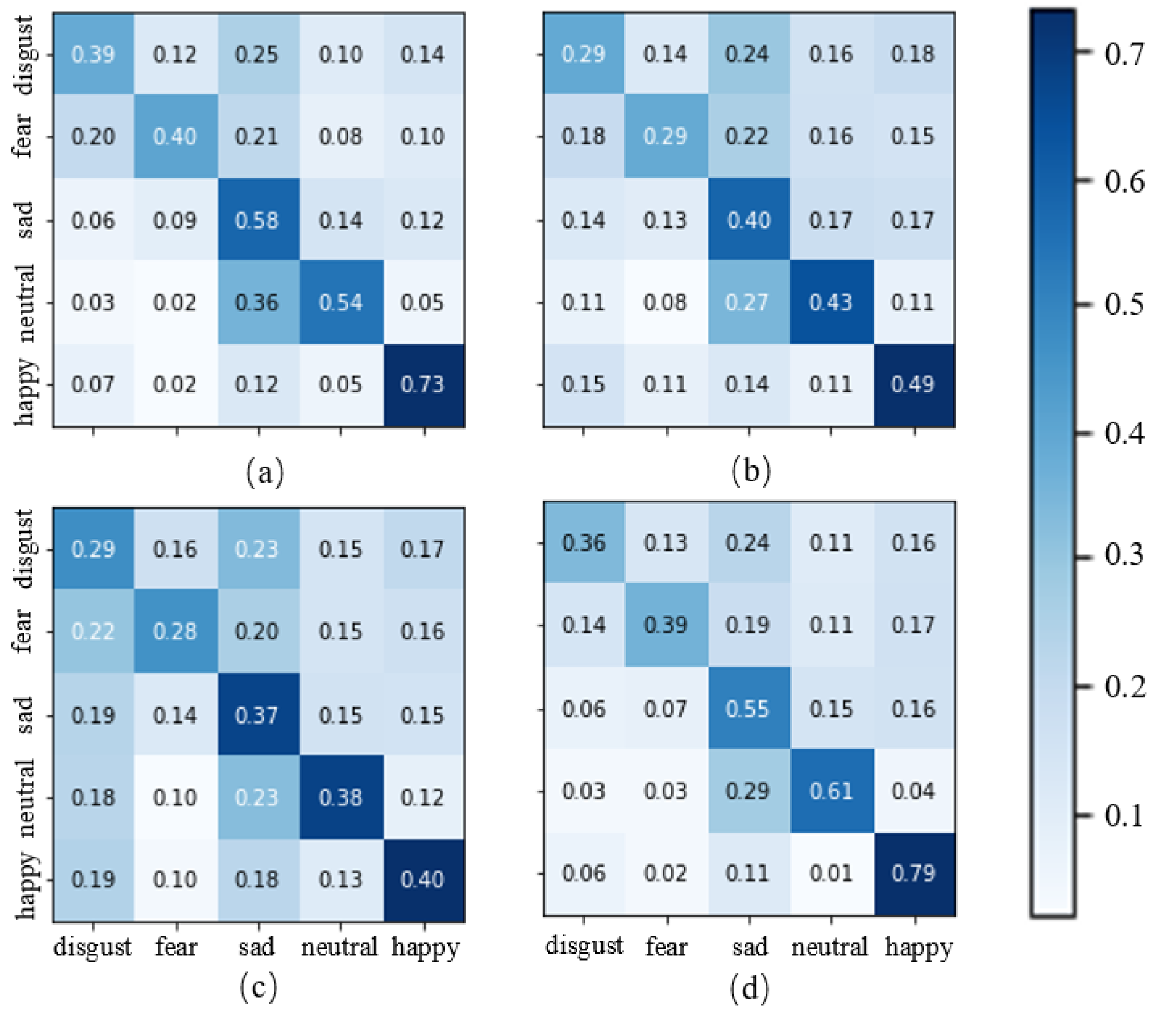

Figure 5.7: The confusion matrices of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream.

I also did subject dependent test and cross session test and compared the accuracy of my proposed methods with SVM, ELM, BDAE and HNSN. Table 5.5 shows the performance comparison between the proposed method and other methods.

5.5 Conclusion

This chapter presents a multi-stream features combination method for EEG-based emotion recognition. The problem is approached from three main directions: 1) Decomposing and transforming one-dimensional EEG signals into temporal and spatial

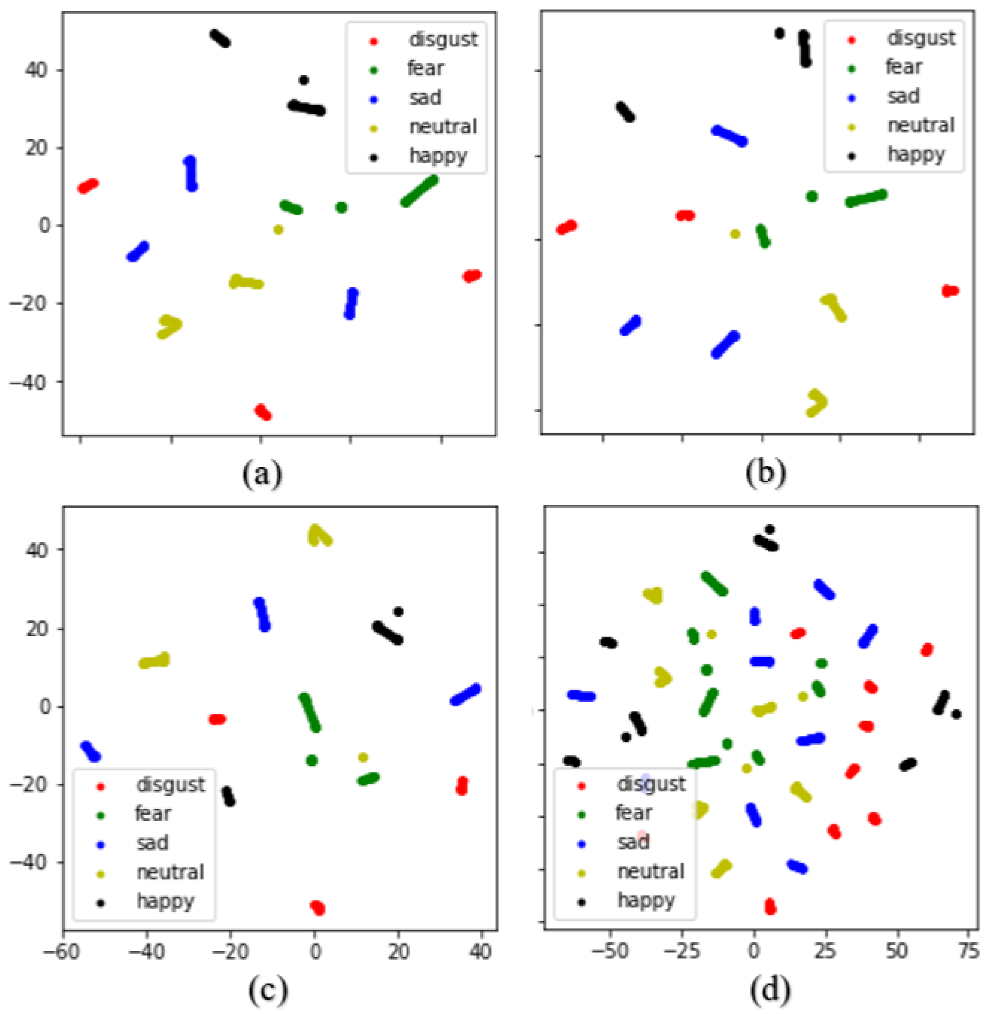

Figure 5.8: The visualization results of each single stream and three stream fusion approaches: (a) Frequency stream. (b) Spatial stream. (c) Temporal stream. (d) Multi-stream.

and frequency features is effective for emotion recognition and 2) features extracted from both nonlinear and linear multi-layer network, rather than a single linear network and 3) Early fusion incorporates multiple morphological features. Experimental results show that my method can be used as a local feature extractor, Combiner and classifier, and its performance is better than other methods.

Chapter 6

Conclusion & Future Work

6.1 Overview 61
6.2 Main Contributions 61
6.3 Conclusion 62
6.4 Future Work 63

6.1 Overview

The approaches proposed in this thesis has shown good overall accuracy with room for improvement. The findings of this research can be used in future for solving similar problems in polynomial time.

6.2 Main Contributions

I verified the importance of feature learning for machine learning through three experiments described in three chapters. In this thesis we have proposed multiple methods to select, construct, extract, and fuse data features. And we have applied my method to one-dimensional and three-dimensional data, and achieved good results.

There are several key contributions of this thesis:

- I propose an algorithm that converts one-dimensional signals with space and time characteristics into two-dimensional space matrix and time matrix.

- Effect of specific channels of DE features: Previous studies [9, 27, 53] have shown that specific channels of DE features may influence the result of EEG-based emotion recognition. Motivated by these experiments, we add weights to the 12 corresponding channel values, while we generate EEG images. After many experiments, we adjusted weight values and obtained a good performance.
- I propose an unsupervised learning method that uses DCNN autoencoder model to extract spatial features. And it has been proved through experiments that this feature can well describe the EEG signal data.
- I propose a multi-stream hierarchical network framework learning behaviors of features combined from multi networks. Each stream can extract temporal, spatial and frequency features respectively, which significantly improve accuracy. I adopted an early fusion method that is better than later fusion by fusing different features into one super vector.
- For EEG-based emotion recognition, my proposed method is based on the extraction of second-level features. The experimental results prove that the second-level features have better ability to describe the input data than the first-level features.
- My method proves that transfer learning is effective in 3D CNNs, and the video features directly extracted through 3D CNNs and used in HNSN-based networks can improve recognition accuracy.

6.3 Conclusion

There are many methods of feature learning in this problem. I can get the following three conclusions:

- Splitting EEG data into multiple stream such as space, time, and frequency, and extracting features separately, can better and more accurately describe EEG signal data from multiple angles.
- Features extracted from both nonlinear and linear multi-layer network, rather than extracted from a single linear network.
- Same with 2D transfer learning, 3D transfer learning is effective.

- Instead of using randomized input weights, we can approach a classifier where weights would be configured by calculation and reach to the steepest descent in iterative manner without configuring the learning rate.
- For training data with multi-dimensional features, splitting them by dimensions and extracting features can effectively improve the recognition accuracy.

6.4 Future Work

In the future, we still have a lot of research to continue to improve and validate my method.

- For EEG emotion recognition, different emotion features are often expressed in different frequency bands, so we plan to test different frequency bands. The impact of data on recognition results, and increase the extraction and fusion of data features in different frequency bands to improve recognition accuracy.
- For video data, we plan to convert the video data into spatial data (i.e. a single video frame) and temporal data (generated by my proposed algorithm). These data are used to extract features respectively, and the features extracted from the original data are fused to test the recognition accuracy.
- I apply my proposed method to other 1D data to verify the generalization performance of my method.

Bibliography

[1] G. L. Ahern and G. E. Schwartz. Differential lateralization for positive and negative emotion in the human brain: EEG spectral analysis. *Neuropsychologia*, 23(6):745–755, 1985.

[2] S. Alhagry, A. A. Fahmy, and R. A. El-Khoribi. Emotion recognition based on eeg using lstm recurrent neural network. *International Journal of Advanced Computer Science and Applications*, 8(10), 2017.

[3] M. Balconi and C. Lucchiari. Consciousness and arousal effects on emotional face processing as revealed by brain oscillations, a gamma band analysis. *International Journal of Psychophysiology*, 67(1):41–46, 2008.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[6] X. Chai, Q. Wang, Y. Zhao, Y. Li, D. Liu, X. Liu, and O. Bai. A fast, efficient domain adaptation technique for cross-domain electroencephalography(eeg)-based emotion recognition. *Sensors*, 17(5), 2017.

[7] G. Chanel, J. J. M. Kierkels, M. Soleymani, and T. Pun. Short-term emotion assessment in a recall paradigm. *International Journal of Human-Computer Studies*, 67(8):607–627, Aug 2009.

[8] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[9] R. Duan, J. Zhu, and B. Lu. Differential entropy feature for eeg-based emotion classification. In *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 81–84, Nov 2013.

[10] 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*, pages 1933–1941, 2016.

[11] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars. Rank pooling for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):773–787, 2016.

[12] S. K. Hadjidimitriou and L. J. Hadjileontiadis. Toward an eeg-based recognition of music liking using time-frequency analysis. *IEEE Transactions on Biomedical Engineering*, 59(12):3498–3510, Dec 2012.

[13] K. Hara, H. Kataoka, and Y. Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[15] 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*, pages 770–778, 2016.

[16] S. Herath, M. Harandi, and F. Porikli. Going deeper into action recognition: A survey. *Image and vision computing*, 60:4–21, 2017.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[18] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2011.

[19] J. J.B.Allen, J. A.Coan, and M. Nazarian. Issues and assumptions on the road from raw signals to metrics of frontal eeg asymmetry in emotion. *Biological Psychology*, 67(1):183–218, 2004.

[20] R. Jenke, A. Peer, and M. Buss. Feature extraction and selection for emotion recognition from eeg. *IEEE Transactions on Affective Computing*, 5(3):327–339, July 2014.

[21] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011.

[22] Q. Li, Z. Qiu, T. Yao, T. Mei, Y. Rui, and J. Luo. Action recognition by learning deep multi-granular spatio-temporal video representation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, pages 159–166, 2016.

[23] T. Li, W. Liu, W. Zheng, and B. Lu. Classification of five emotions from eeg and eye movement signals: Discrimination ability and stability over time. In *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 607–610, March 2019.

[24] Y. Lin, C. Wang, T. Jung, T. Wu, S. Jeng, J. Duann, and J. Chen. Eeg-based emotion recognition in music listening. *IEEE Transactions on Biomedical Engineering*, 57(7):1798–1806, July 2010.

[25] A.-A. Liu, Y.-T. Su, W.-Z. Nie, and M. Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(1):102–114, 2016.

[26] J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European conference on computer vision*, pages 816–833. Springer, 2016.

[27] Y. Lu, W.-L. Zheng, B. Li, and B.-L. Lu. Combining Eye Movements and EEG to Enhance Emotion Recognition. *International Joint Conferences on Artificial Intelligence Organization*, pages 1170–1176, 2015.

[28] D. Mantini, M. G. Perrucci, C. Del Gratta, G. L. Romani, and M. Corbetta. Electrophysiological signatures of resting state networks in the human brain. *Proceedings of the National Academy of Sciences*, 104(32):13170–13175, 2007.

[29] S. Nah, T. Hyun Kim, and K. Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[30] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017.

[31] M. Rigotti, O. Barak, M. R. Warden, X.-J. Wang, N. D. Daw, E. K. Miller, and S. Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, pages 585–590, 2013.

[32] D. Sammler, M. Grigutsch, T. Fritz, and S. Koelsch. Music and emotion: Electrophysiological correlates of the processing of pleasant and unpleasant music. *Phychopysiology*, 44(2):293–304, 2007.

[33] M. Sarlo, G. Buodo, S. Poli, and D. Palomba. Changes in eeg alpha power to different disgust elicitors: the specificity of mutilations. *Neuroscience Letters*, 382(3):291–296, 2015.

[34] A. Schaefer, F. Nils, X. Sanchez, and P. Philippot. Assessing the effectiveness of a large database of emotion-eliciting films: A new tool for emotion researchers. *Cognition and Emotion*, 24(7):1153–1172, 2010.

[35] B. Schuller. Recognizing affect from linguistic information in 3d continuous space. *IEEE Transactions on Affective Computing*, 2(4):192–205, Oct 2011.

[36] L. Shi, Y. Jiao, and B. Lu. Differential entropy feature for eeg-based vigilance estimation. In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 6627–6630, July 2013.

[37] L. Shi and B. Lu. Off-line and on-line vigilance estimation based on linear dynamical system and manifold learning. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 6587–6590, Aug 2010.

[38] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[39] T. Song, W. Zheng, P. Song, and Z. Cui. Eeg emotion recognition using dynamical graph convolutional neural networks. *IEEE Transactions on Affective Computing*, pages 1–1, 2018.

[40] W. Song, J. Yu, X. Zhao, and A. Wang. Research on action recognition and content analysis in videos based on dnn and mln. *Computers, Materials and Continua*, 61(3):1189–1204, 2019.

[41] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[42] J. Tang, C. Deng, and G. Huang. Extreme learning machine for multilayer perceptron. *IEEE Transactions on Neural Networks and Learning Systems*, 27(4):809–821, April 2016.

[43] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia. Detail-revealing deep video super-resolution. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[44] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[45] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1510–1517, 2017.

[46] L. Wang, Y. Qiao, and X. Tang. Mofap: A multi-level representation for action recognition. *International Journal of Computer Vision*, 119(3):254–271, 2016.

[47] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[48] Y. Yang, Y. Wang, and X. Yuan. Bidirectional extreme learning machine for regression problem and its learning effectiveness. *IEEE Transactions on Neural Networks and Learning Systems*, 23(9):1498–1505, 2012.

[49] Y. Yang and Q. M. J. Wu. Extreme learning machine with subnetwork hidden nodes for regression and classification. *IEEE Transactions on Cybernetics*, 46(12):2885–2898, Dec 2016.

[50] Y. Yang, Q. M. J. Wu, and Y. Wang. Autoencoder with invertible functions for dimension reduction and image reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(7):1065–1079, July 2018.

[51] Y. Yang, Q. M. J. Wu, W. Zheng, and B. Lu. Eeg-based emotion recognition using hierarchical network with subnetwork nodes. *IEEE Transactions on Cognitive and Developmental Systems*, 10(2):408–419, June 2018.

[52] D. Yuan, G. Shan, T. Kun, L. Jiqing, and W. Haila. Performance evaluation of early and late fusion methods for generic semantics indexing. *Pattern Analysis and Applications*, 17(1):37–50, Feb 2014.

[53] W. Zheng and B. Lu. Investigating critical frequency bands and channels for eeg-based emotion recognition with deep neural networks. *IEEE Transactions on Autonomous Mental Development*, 7(3):162–175, Sep. 2015.

[54] W. Zheng, J. Zhu, and B. Lu. Identifying stable patterns over time for emotion recognition from eeg. *IEEE Transactions on Affective Computing*, pages 1–1, 2018.

[55] W. Zheng, J. Zhu, Y. Peng, and B. Lu. Eeg-based emotion classification using deep belief networks. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2014.