

國立交通大學

電機與控制工程學系

碩士論文

腦神經訊號源自動分類系統

**Automatic Classification System
of Useful EEG Source**

研究生：黃華山
指導教授：林進燈 教授
周志成 教授

中華民國九十八年八月

腦神經訊號源自動分類系統

**Automatic Classification System
of Useful EEG Source**

研究生：黃華山

Student: Jui-Hsin Tung

指導教授：林進燈 教授

Advisor: Prof. Chin-Teng Lin

周志成 教授

Prof. Chi-Cheng Jou



A Thesis

Submitted to Department of Electrical and Control Engineering
College of Electrical Engineering
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of Master
in
Electrical and Control Engineering
August 2009
Hsinchu, Taiwan, Republic of China

中華民國 九十八年 八月

腦神經訊號源自動分類系統

學生：黃華山

指導教授：林進燈 博士

周志成 博士

國立交通大學電機與控制工程研究所

中文摘要

腦波訊號的雜訊去除在研究上或應用上是一個很重要的步驟，在腦波訊號上的雜訊可能會帶有眼動、肌肉信號、交流電干擾以及測量平台的環境所造成的。這些雜訊會使得腦波信號會失真而無法使用，或是誤解是新的現象而讓預測結論無法預期。由近幾年的期刊上常見獨立成份分析，是一個方便的訊號分離的方法，能使得我們研究時能從分離出來訊號在頭殼上的位置，來辨別此獨立訊號在頭殼上所扮演的角色，以及此獨立訊號是否帶有嚴重的雜訊。如果需要還原為腦波訊號，能將帶有高度雜訊的獨立訊號去除後還原回原本腦波訊號的時域。從過去已有期刊能證明此方法有高度的雜訊去除能力，但是目前此方式僅只於依照個人經驗的選擇，尚未有標準化的分離訊選擇標準，使得腦機介面或即時腦波應用上目前尚無法以此方法提供良好的腦波訊號。在這篇論文裡，我們以基礎的計算機智慧的學習方式能證明，能夠以基礎的方法來分離出帶有高度雜訊的獨立訊號，因此也證明製作自動化的獨立訊號分析雜訊去除系統是能實現的方式。

關鍵字：腦電波、獨立成份分析、機器學習

Automatic Classification System of Useful EEG Source

Student: Hwa-Shan Huang

Advisor: Dr. Chin-Teng Lin
Dr. Chi-Cheng Jou

Department of Electrical and Control Engineering
National Chiao Tung University

Abstract

Removal of artifacts is an important step in any research or application of electroencephalogram (EEG). The artifacts may contain eye-blinking, muscle noise, heart signal, line noise, and environmental effect. Such noises often make the raw EEG signals not very useful for extraction/identification of physiological phenomena from EEG. The independent component analysis (ICA) is a popular technique for artifact removal in brain research and some reports demonstrate that ICA can remove the artifacts with lower (acceptable) loss of information. However, these reports select useful independent components manually, primarily by looking at the scalp-plots. This is of great inconvenience and is a barrier for BCI or real-time applications of EEG. In this thesis, we demonstrate that machine learning methods could be quite effective to discriminate useful independent components from artifacts and our findings suggests the possibility of developing a ‘universal’ machine for artifact removal in EEG.

Keyword: EEG, Independent Component Analysis (ICA), Machine Learning

致 謝

Acknowledgement

在本論文完成的同時,最先感謝的是我的指導教授 林進燈教授與 周志成教授. 林進燈教授在過去兩年的研究期間, 提供各種的研究資源以及良好的實驗環境, 並常在我研究中迷路時指點方向. 周志成教授點出在我分析上的盲點, 使我能修正研究中尚未注意到的缺失. 感謝兩位老師的助益能使我論文能順利完成.

我還要感謝在印度 Indian Statistical Institute 的 Professor Pal 以及柯立偉博士在分析上的指導與監督. 感謝 Prof. Pal 令人驚奇的奇妙點子, 讓我本篇論文能有重要的起頭, 以及提攜我分析方法的構思. 感謝柯立偉博士不厭其煩的理解我不清晰的表達, 關心我研究的近況, 以及教導各種分析的意義. 感謝 Prof. Pal 及柯立偉博士一路帶領我學習訊號分析, 實驗數據統整還有負責的態度, 這對於我本身是重大的自我改進過程, 以及是一生中難忘的經驗.

也感謝我的家人, 在我辛勞時, 能包容我專注在程式上而缺少聯繫, 在我疲累時, 能迎接我回家盡情休憩. 忍受我的任性, 包容我的軟弱, 使我能支持到論文寫完的一刻, 我都要感謝有家人心靈上的支持. 也還要感謝實驗室的同學與朋友, 在我心煩意亂的時候, 為我打氣, 在我過多課業時, 能一起分憂. 沒有實驗室大家的幫忙, 不管在修課上, 生活上, 都會缺少共同奮鬥的樂趣, 以及共同腦力激盪的娛樂.

僅以本文獻給我親愛的朋友, 關心我的家人, 以及照顧我的師長, 願大家能共享我的喜悅與感恩.

Contents

Chinese Abstract	i
English Abstract	ii
Acknowledgement	iii
Contents	iv
Figures	vi
Tables	vii
1. Introduction	1
1-1. Property of EEG Signals	1
1-1-1. Mixing Signals.....	2
1-1-2. Noise Artifacts.....	3
1-2. Motivation	6
1-2-1. EEG Feature Extraction.....	6
1-2-2. Literatures Survey.....	7
1-3. Problems.....	12
1-4. Goals of this Study	15
1-5. Organization of this Study.....	16
2. Data Acquisition and Pre-Processing	18
2-1. EEG Data Collection.....	18
2-2. Subjects	20
2-3. Independent Component Analysis	20
2-4. Weight Matrix Normalization	23
3. Automatic Scheme for Useful Independent Component Selection	26
3-1. Labeling of Useful / Useless Components	26
3-2. Supervised Methods	31
3.2.1. Multilayer Perceptron Neural Network.....	31
3.2.2. Radial Basis Function Neural Network.....	34
3.2.3. Support Vector Machine	36
3-3. Training Algorithm Structure.....	38
3-4. Cross Validation.....	40
3-5. Testing Algorithm Structure.....	41
4. Experiment Results	44
4-1. Cross Validation Results of Training Data	44
4-2. Evaluation Performance of Testing Data	46

5. Discussion	52
5-1. The Difference between Training and Testing.....	52
5-2. The Optimal Threshold	52
5-3. The Positive Predictive Rate (PPR)	53
6. Conclusion	54
7. Future Works	56
7-1. The Enhancement of Feature.....	58
7-2. Auto Clustering System	59
Reference	60
Appendix.....	66



Figures

Figure 1-1. The electrode records mixing brain activities.	3
Figure 1-2. The noise signals.	4
Figure 1-3. A diagram of EEG analysis by using ICA.	14
Figure 1-4. An automatic EEG analysis scheme by using ICA.	14
Figure 2-1. The 32 channel EEG cap and electrodes placement of international 10–20 system.....	19
Figure 2-2. An example of removing artifacts by ICA.	22
Figure 2-3. An example of weight vector.	24
Figure 2-4. An illustration of how a scalp map is drawn.	24
Figure 2-5. A central component scalp map’s cross section.	25
Figure 3-1. A set of illustrative ICA scalp map labeled by experts.	28
Figure 3-2. An example of one hidden layer neural network architecture....	32
Figure 3-3. Ten-fold cross-validation of training algorithm structure.	40
Figure 3-4. Testing algorithm through ten-fold cross validation training classifiers.....	43
Figure 4-1. Average accuracy of testing datasets from 10 subjects via different threshold score.....	48
Figure 4-2. Average useful component predictive rate from 10 subjects via different threshold score.....	50
Figure 7-1. An illustration of how a scalp map is drawn.	57

Tables

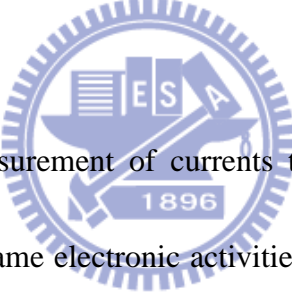
Table 3-1. An example of useful component set.....	29
Table 3-2. A example set of useless components.....	30
Table 4-1. The accuracy of 10-fold cross validation.....	45
Table 4-2. The parameters of supervised methods.....	45
Table 4-3. Classification accuracy of each subject under threshold 0.6.	48
Table 4-4. Useful component predictive rate of each subject under threshold 0.6.....	50



1. Introduction

This thesis presents an automatic method for artifacts removal in independent component analysis (ICA), this is a new implementation for extending the ICA application. At the first, we introduce what Electroencephalography (EEG) is and the noisy signals in EEG.

1-1. Property of EEG Signals



An EEG signal is a measurement of currents that flow between the neurons (nerve cells), which have the same electronic activities from a small zone in the gray matter of brain. We use electrode to collect one channel EEG signal, and the multichannel recording with electrodes, electrode caps are often used. Thus, we could collect the EEG signal to analyze the specific pattern on EEG activity based on subject behaviors. However, the EEG signal is recorded from electrode cap. Such structure leads the electrodes which collect the multiple zones of EEG activities and noises.

1-1-1. Mixing Signals

EEG signals are resulted from the magnetic field variation caused by the compound delivery in the brain cell. We can measure the minor field variation on the scalp, and then after amplifying the signals we can record them as the so-called EEG signals. The EEG signals will measure operation in the various areas in cortex as shown Fig. 1-1. Therefore, EEG signals are mixed signal, they will mix from each source in the gray matter of brain. However, the transmission distance will result in the decayed performance while receiving the signals. In Fig. 1-1, the signals measured from the left and right will be different due to the different transmission distance. Like Channel(A) has high amplitude from Source(A) and lower from Source(B), the signal amplitude is depend on the inverse proportion of distance .Therefore, like in a noisy meeting, we can separate the noises from the language voice. The noisy signal could be take out the EEG signals, too.

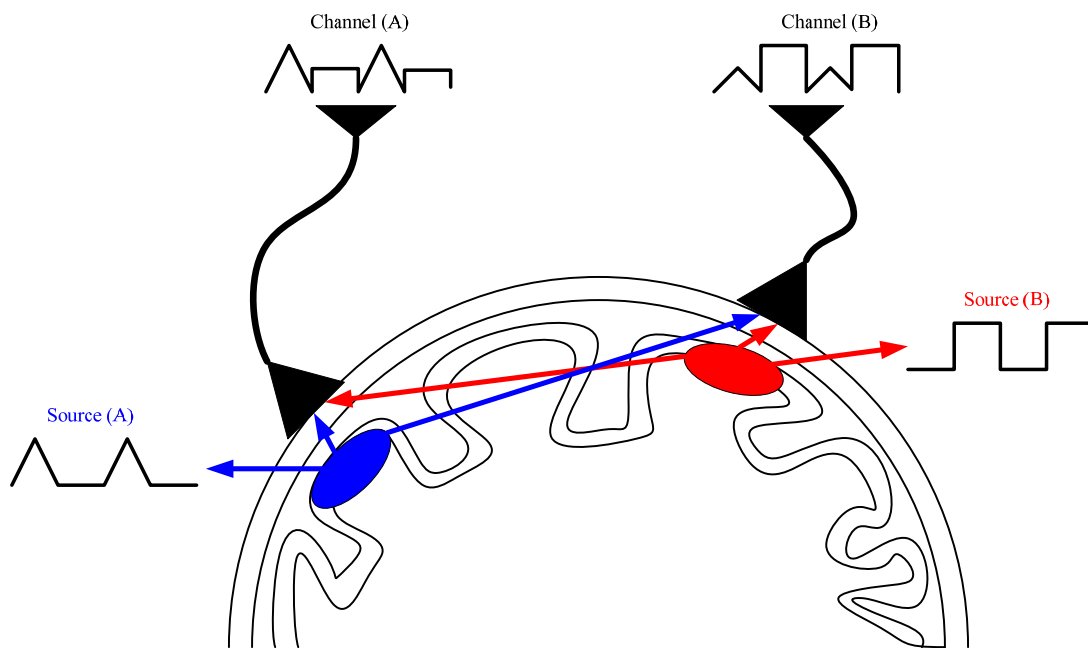


Figure 1-1. The electrode records mixing brain activities.



1-1-2. Noise Artifacts

The noises are not always acceptable in all measured systems because it perhaps will make the signals with noises unusable. In EEG signals processing, we will eliminate the noises perturbation as possible as we can. Since we couldn't ensure the perturbation signals waveform from the original EEG signals, the normal variation could be a characteristic for analysis. However, the perturbation of noises will destroys or covers the original existed characteristics such that the EEG signals cannot be applied to phenomenon analysis or BCI applications.

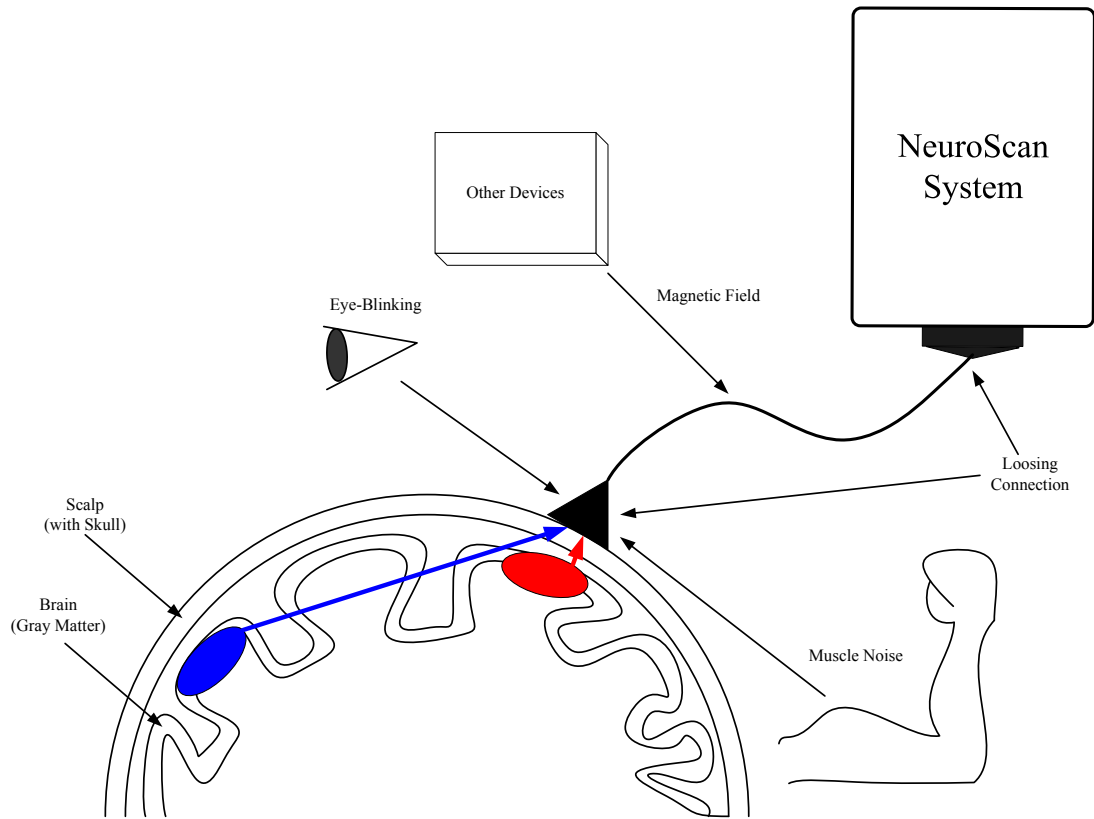
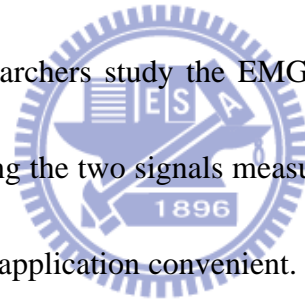


Figure 1-2. The noise signals.



EEG signals measurement can be divided into two classes. The First is external noises. The external noises will be affected by the instability of external environment and measurement interface. Such in Fig. 1-2, the device will produces line noises due to the interfering of magnetic field. This frequency, about 60 Hz, can be filtered by low pass filter. However, the bad interface contact will be an important problem between scalp and electrodes. Though we will ensure the measurement impedance under 5K, the testers will also unintentionally wave their heads during the testing, which will make the bad interface contact and the bad signals. While we encounter

such problem, we can only remove all the periods of bad signals. The problems of external noises in measurement can be improved by reforming experiment quality to get better performance. The second is internal noises. The internal noises mainly come from the eye movement[1] and muscle signals[2, 3]. In Fig. 1-2, such noises will be absorbed in electrodes while measuring electromyography (EMG) signals. They may not be filtered or cleaned by enhancing experiment quality. Such biomedical signals will also be transferred to electrodes through scalp. For eye movement, some researchers used the eye movement equipment to eliminate the eye movement signals. Some researchers study the EMG elimination. However, for the real-time BCI system, involving the two signals measurement will result in the line in BCI too long to make the real application convenient. Therefore, the direct analysis to distinguish the EEG signals from the internal noises not only can solve the internal noises problem, but also can make BCI application broader, that is to say, it will not be limited by only measuring the minor noises influenced electrodes. In the following, we will introduce the study about solving this problem.



1-2. Motivation

Removing the artifacts in electroencephalogram (EEG) signal is an important but difficult issue in EEG based applications. However, there are many kinds of reasons to produce large and irregular artifacts in EEG recording, such as eye-blinking, muscle noise, heart signals, line noise and environment factors, etc. The line noise and environment factors are external noise, which is controlled by experiment and this may be neglected by increasing signal to noise ratio. But, the eye-blinking, muscle noise and heart signal are internal noise, which is always existence and influence raw EEG signal. If noises like eye-blinking appear frequently, the EEG signal would not be very useful. For EEG based research, it requires an automatic method that can preserve the EEG signal while removing the artifacts.

1-2-1. EEG Feature Extraction

The EEG noisy signals have the regular form like the eye-blinking and electrocardiography (ECG), and the irregular form like the muscle activities and eye-saccade in time course. For the regular noise, we can remove it by subtracting the

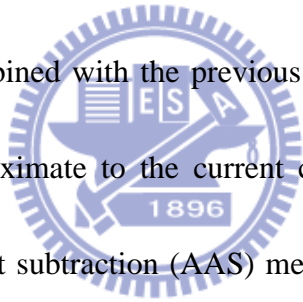
reference from eye-tracking and ECG, or find the similar pattern to remove it. However, the irregular noises like high frequency activities could be eliminated by low pass filter, but the other irregular noises are hardly weakened. Therefore, the mathematical source separation method is a good approach to solve this problem.

1-2-2. Literatures Survey

It is known that there are many artifacts removing methods. There are three types we could figure: First, Find out the error signal model, and then eliminate all the similar models along the time course. Second, the noises have high frequency that we could filter it by low pass filter in frequency course .Third, we assume the noises are mixed in the original signal, we could separate them into different instance. Then we can divide them by signals processing methods. We simply introduce them well known as follows:

(a). Raw EEG Analysis

The EEG analysis has many approaches [3-10] to eliminate the artifacts, and we introduce some common methods in researches. First, the time-domain regression [8, 10] is introduced. The time-domain regression does the linear regression analysis between the reference signal (EOG) and the EEG signal to perform a regression parameter. Then the EEG signal subtracts the reference signal plus the regression parameter to get the approximate solution. Second, the adaptive filtering method [7-10] is considered. The adaptive filtering needs a reference signal, which usually is EOG as the reference to remove the ocular artifacts. Then, in EEG ocular artifact cancellation, the EOG is combined with the previous clean EEG signal to the finite response filter (FIR) to approximate to the current clean EEG signal. The second method is the averaged artifact subtraction (AAS) method [4-6], which finds similar artifact peaks and averages these artifacts to generate a subtraction model. This is then subtracted from the noisy EEG to get the clean EEG signals [5, 6]. However, these methods only can process the artifacts with fixed feature like EOG. The noisy signals like muscle noise don't exist any feasible pattern for these method, the research present the detection of muscle noise by the power in 26.25~32.0Hz [3]. Furthermore, because the filtering method needs a reference signal, but we only pay the attention on



the EEG signals. Thus, we only discuss the AAS method with other method in below sections.

(b). Principle Component Analysis

The first introduction of blind source separation is principle component analysis (PCA). The PCA translates the original data into some important components, which mix the origin signals from covariant matrix. Therefore, we could receive a linear transform matrix after PCA, then column of inversed weight matrix could be present on the scalp map depending on the channel location [11, 12]. PCA based methods result in components where a single component may contain too much information (signal). Furthermore, the muscle noise could be reduced by separating into different components. However, the ocular is a significant signal in PCA components, which mix large signals from EEG. Thus, if we remove the ocular component from PCA, there are many signals would be removed with the ocular component and distorted the back-projection signal [13, 14]. Therefore, the PCA method could reduce the muscle noise, but the ocular effect may not do a good job.

(c). Independent Component Analysis

The second introduction of blind source separation is independent component analysis (ICA). Among the different methods of artifacts removal, ICA [15] is a frequently used method. ICA methods generally assume that the signals recorded on the scalp are mixtures of time courses of temporally independent cerebral and non-cerebral sources. The potentials generated by different parts of the brain, scalp, and body are linearly summed at the recording electrodes. ICA also assumes that the propagation delays are not significant. Many investigations have demonstrated that ICA could separate the artifacts and raw EEG signals [13, 15-22]. Jung et al. in [13] proposed a method for isolating and removing different types of EEG artifacts by linear decomposition using an extension of the Bell & Sejnowski,'s information-maximization ICA algorithm [23, 24]. The extended algorithm [24] can separate sources with either super-Gaussian or sub-Gaussian amplitude distributions. This enables one to remove line noises efficiently. The algorithm does not require reference channels for the artifact sources. The method first finds the independent time courses of different cerebral and artifact sources and then the cleaned EEG signals are obtained by eliminating the contributions of the artifact sources. Authors

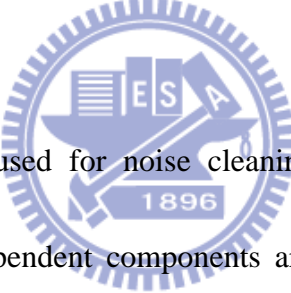
in [16, 17, 20, 22] have used ICA to remove the effect of eye-blinking. The EOG can be used to record the artifacts caused by eye-blinking in EEG signal. Then a linear combination of EOG signal recorded at different sampling time can be subtracted to find the artifact free EEG. Also, the EOG signal can be compared with the independent components to identify independent components relating to EOG related artifacts [16].

Independent component analysis has been effectively used in removing artifacts from fMRI also [25-27]. For example, task related motion is a significant source of motion in fMRI. Kochiyama et al. [26] used an IC based method to remove the effect of task related motion. Then they remove the ICs associated to task related motion and reconstruct the image. McKeown et al. [27] made an interesting application of ICA and have demonstrated that ICA can very effectively be used to separate fMRI data into meaningful constituent components. It can also separate consistently and transiently task-related physiological changes, nontask-related physiological phenomena, as well as movement artifacts. This researches evidence that ICA has good results in source separation experiments.

The comparison of raw EEG analysis and ICA, AAS method cannot rule out the possibility of inclusion of some useful actual EEG signal in the subtraction model,

and consequently it may induce distortions on scalp maps [4]. This reference also shows that the ICA has better quality of artifact removed EEG than AAS. The comparison of PCA and ICA in [13, 14, 28-31] shows that ICA performs better resolution. The artifact components extracted from ICA are generally found to exhibit stronger correlation with the actual artifacts than that by the PCA artifact components. All of these have motivated us to consider ICA based artifact removal from EEG.

1-3. Problems



Typically when ICA is used for noise cleaning in EEG signal we use the following steps: first the independent components are computed, then the artifacts (some ICs) are removed and then EEG signal is reconstructed using the useful (good) ICs by projecting back (Fig. 1-3). But, unfortunately these researches select the useful (good) independent components manually [13, 16-18, 20, 21, 32], usually by looking at the scalp maps. Such a system has several drawbacks: First it is dependent on experts. So the availability of experts and subjective judgment of experts could be important determinant of the success of any application based on such ICs. The neuroscientist and psychologist would be inconvenience when be introduced in ICA

analysis. And, they would get big trouble on noisy components without the standard of component selection. Second, for many real-time applications such as detection of drivers' drowsiness, or in some BCI applications, we cannot use such methods. These problems can be eliminated if we can do the job of selecting useful ICs in an automatic manner. Thus, we want to develop a learned machine, which, given the ICs (in other words given the mixing matrix), can select the useful ICs in an automatic manner. Such a system can make BCI applications standalone (free from human intervention) and can facilitate real-time applications. This also opens up the possibility of developing a "universal" EEG artifact removal system.



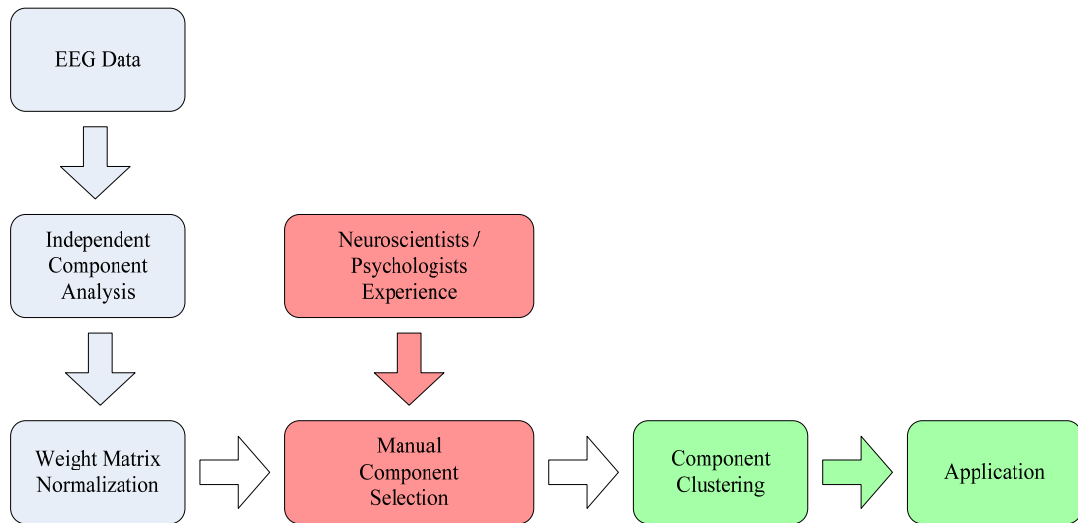


Figure 1-3. A diagram of EEG analysis by using ICA.

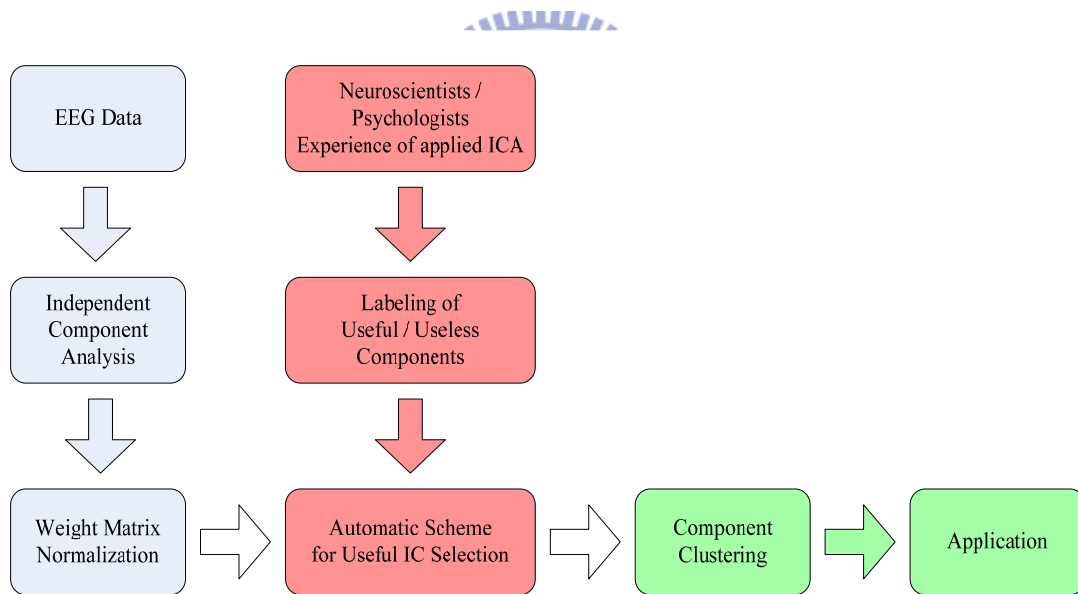


Figure 1-4. An automatic EEG analysis scheme by using ICA.

1-4. Goals of this Study

We want to implement a new EEG analyzing scheme with ICA, and the Fig. 1-4 is our architecture. The automatic scheme needs a exist labeling database, which from the experience of the expert of neuroscientist. We can use this scheme to analyze the components quickly and feedback to our database form the advice from everyone. Therefore, we need to examine the possibility of implementing the system carefully.

This thesis attempts to demonstrate that it is feasible to develop such a “universal” machine. Here, we use four machine learning tools, the usual multilayer perceptron (MLP) architecture with hyperbolic tangent signal function (MLPTAN) and MLP architecture with redial basis function signal function (MLPRBF), the radial basis function neural network (RBFNN), and the support vector machine with radial basis kernel function (SVMRBF). This four supervised methods are fundamental and wide applying in researches. Some results of this investigation have been reported in [33]. To improve the performance of the proposed method we use the simple majority voting fusion scheme. To demonstrate the effectiveness of such a system we use a 10-fold cross validation protocol to check the consistency of train data, and a 10-classification test to examine the test data accuracy.

1-5. Organization of this Study

In this investigation we introduce a 10-Fold cross validation to check the consistency of feature, and test the classification performance in practice. We shall evident this model is exist. A part of this investigation has been report in [33]. The proposed approach can account for the variability of useful scalp-map between individuals. We need a labeled training data set with information to teach the classifiers the useful / useless scalp-map. In our approach, we label the useful / useless mark on each scalp-map, and assume the useful scalp-map has general feature and the possibility of classification. We divide the train data into 10 folds to find the consistency, and train 10 classifiers to evident the classification performance. We find that, the scalp-map data set exhibit a non-linear classifying group, and the supervised methods have good performance in classification results. Consequently, an automatic selection system can be used to classify useful / useless scalp-maps and that is what we do in this investigation.

This thesis is organized as follows. Chapter2 describes the EEG data collection, Data pre-processing, independent component analysis theory, the scalp-map, and the weight matrix normalization. Chapter3 shows the evaluation of automatic selection

system including 10-Fold cross validation, 10 classifier testing and the threshold of testing results. Chapter 4 shows the experimental results and the discussion is given in chapter5. Finally, we conclude our findings in chapter6.



2. Data Acquisition and Pre-Processing

2-1. EEG Data Collection

We have collected the EEG data from the virtual reality (VR) environment [34-36] installed at the Brain Research Center of the National Chiao Tung University. The EEG data are collected when each subject performs a driving task which is subjected to disturbances. We have used a 32-channel 10-20 system (Fig. 2-1) for recording the EEG, and the impedance between EEG electrodes and skin was kept to less than $5k\Omega$ by injecting NaCl based conductive gel. Data were amplified and recorded by the Scan NuAmps Express system (Compumedics Ltd., VIC, Australia) shown in Fig. 2-1, a high-quality 40-channel digital EEG amplifier capable of 32-bit precision sampled at 1000 Hz. Table 2-2 shows the specifications of the NuAmps amplifier. The EEG data were recorded with 16-bit quantization levels at a sampling rate of 500 Hz in this study. To reduce the burden of computation, the data are then down-sampled to sampling rate of 250 Hz.

However, we recorded the data using 30 channels (there are 2 reference channels). Of these 30 channels, we have ignored 2 channels which are near the eye

position (Fp1, Fp2) to reduce the effect of blinking of eyes. In this investigation the independent components are generated using the EEGLAB platform [37] developed in MATLAB (The Math Works, R2007a). The EEG data were processed using a low-pass filter with a cut-off frequency of 50 Hz in order to remove the power line noise (60Hz) and other high-frequency noise. Similarly, a high-pass filter with a cut-off frequency at 0.5 Hz was applied to remove baseline drifts.

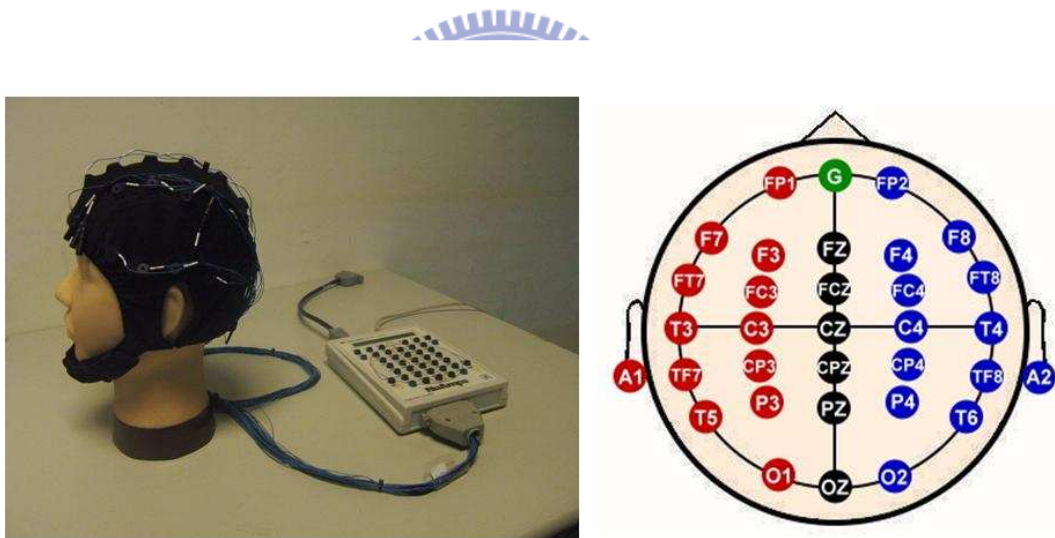


Figure 2-1. The 32 channel EEG cap and electrodes placement of international 10–20 system. The letters used are: F: Frontal lobe. T: Temporal lobe. C: Central lobe. P: Parietal lobe. O: Occipital lobe. "Z" refers to an electrode placed on the mid-line (zero).

2-2. Subjects

We have used a set of 35 subjects (ages varying from 20 to 40 years old) to generate data for the investigation. Subjects were instructed to lane keeping driving experiment. It keeps the car at the center of the lane by controlling the steering wheel, and to perform the driving task consciously in VR environment. The whole driving experiment lasted about 60 to 90 minutes.



2-3. Independent Component Analysis

The independent component analysis [38, 39] is a statistical technique which can separate mixed signals, including the EEG signal, to maximally independent components (ICs) by a specified measure of statistical independence. The ICA assumes a data model $U = WX$, where X is queued column vectors of data recorded from individual EEG channels, W is a weight matrix for mixing independent components back to original signals, U is queued column vector of statistically independent components. In Fig. 2-2, this is the artifact removal method by using ICA.

First, we have a set of collected EEG data, X . Then we implement the ICA algorithm to separate X into component domain (U), and we have the weight matrix W which present the linear combination of each EEG data. Then, we use the back-projection, $X = W^{-1}U$, this method can construct the data from components to origin time domain. We could use columns of W^{-1} to construct ICs as a weighted combination of X . The EEGLAB provides a tool for the visualization of a column of W^{-1} , which is known as component scalp map.



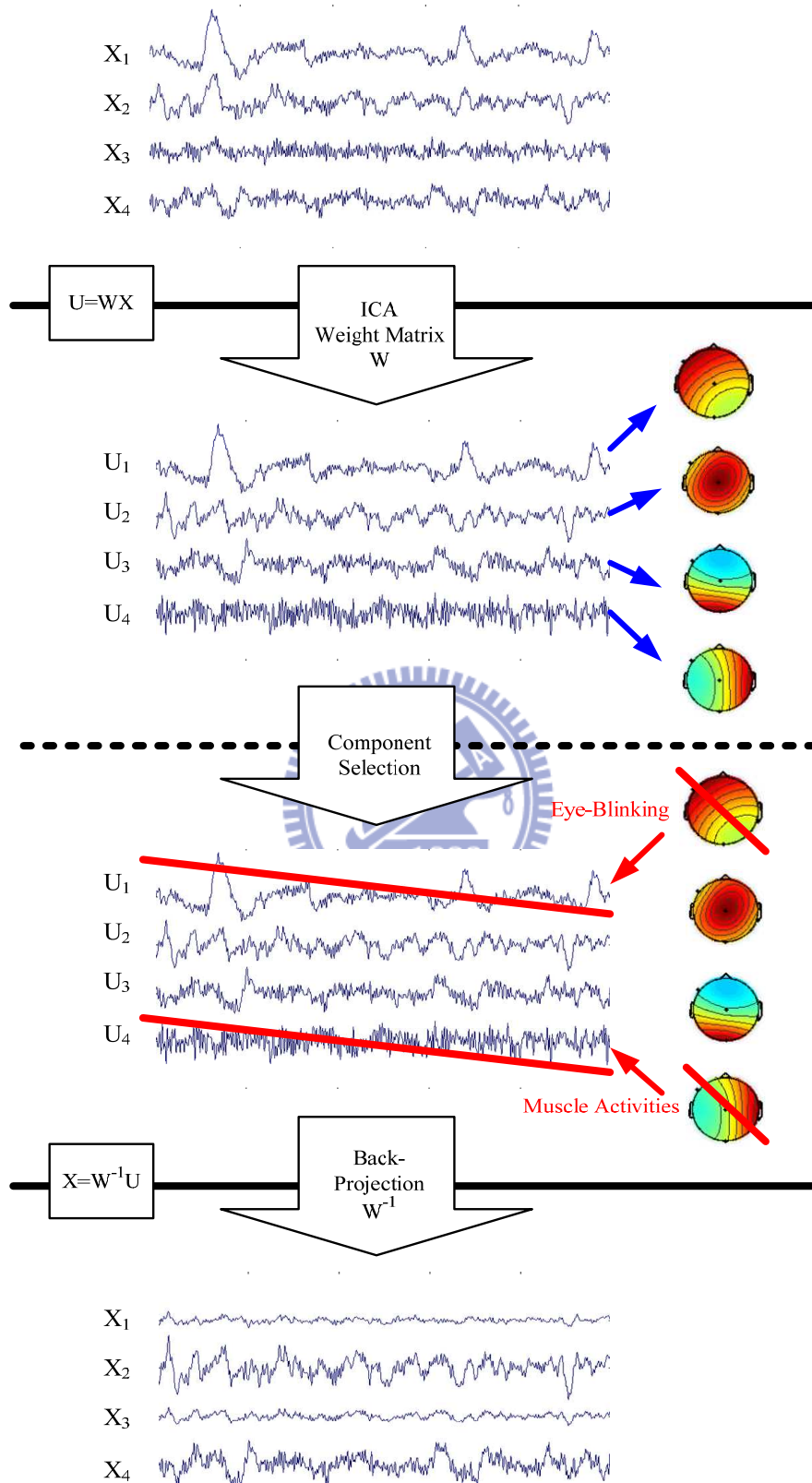


Figure 2-2. An example of removing artifacts by ICA.

2-4. Weight Matrix Normalization

After implement the ICA process, we get the weight matrix. The rows of the weight matrix represent the EEG channels location and the columns of weight matrix represent the ICs, like (Fig. 2-3) [18]. In order to eliminate the different between individuals, we normalize the weight matrix by z-score method. Every component results in a scalp map. We generate the component scalp map using the EEGLAB that uses the value of a normalized weight vector as intensity to fill the relative channel location on the scalp as displayed in Fig. 2-4. The left panel shows the element of a column vector mapped on the scalp location, while the right panel shows the corresponding scalp map. And the weight distribution on scalp-map is like Fig. 2-5. The scalp map depicts projections of the ICs on the electrode sites. The scalp maps help in localizing the activation of the ICs. With the help of scalp maps we can detect the ICs whose activities are concentrated at a particular electrode location.

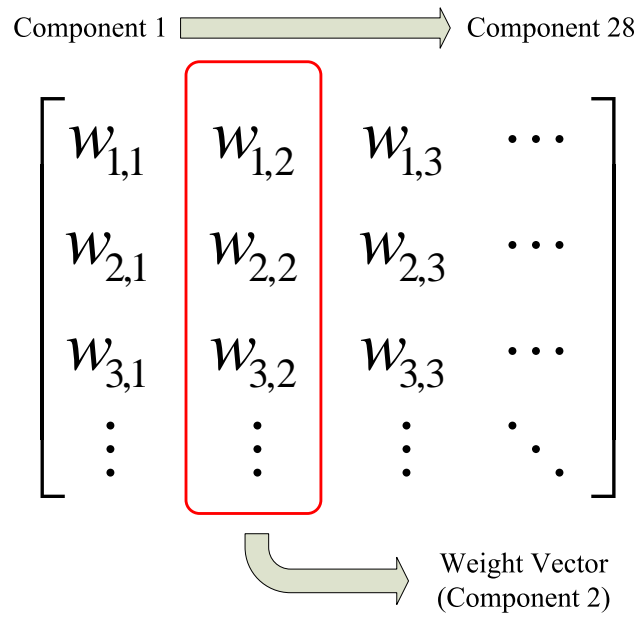


Figure 2-3. An example of weight vector.

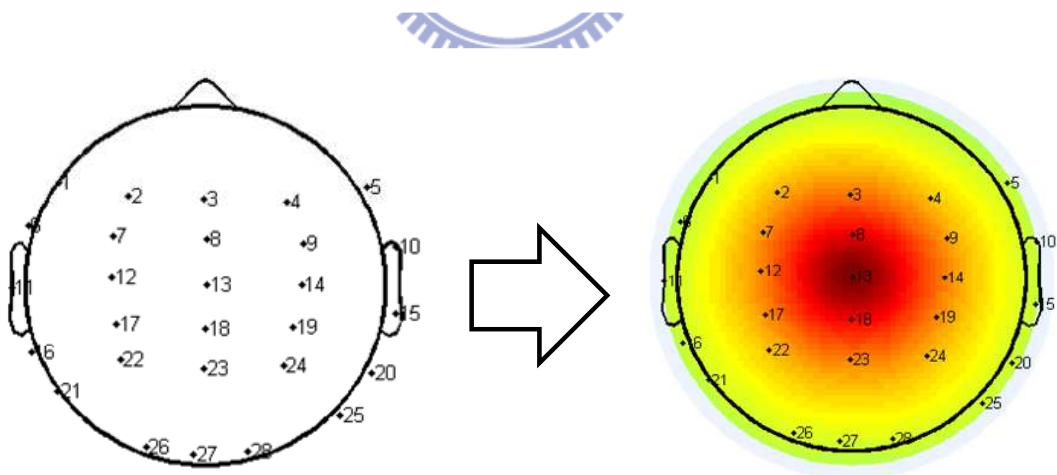


Figure 2-4. An illustration of how a scalp map is drawn.

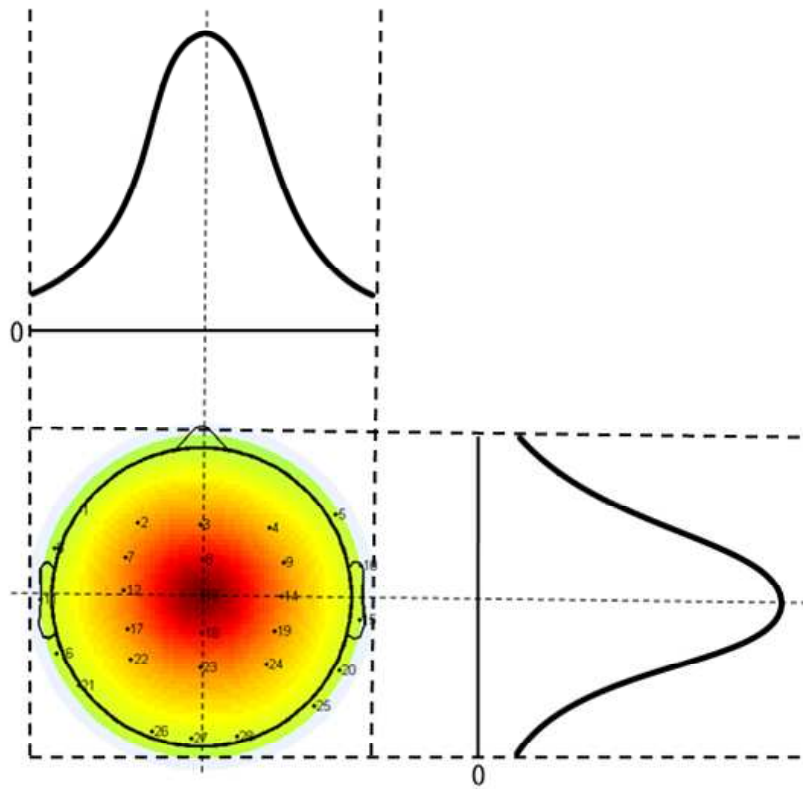


Figure 2-5. A central component scalp map's cross section.

3. Automatic Scheme for Useful Independent Component

Selection

3-1. Labeling of Useful / Useless Components

In almost all applications, an expert looks at such scalp-maps and decides on whether a scalp map (hence an independent component) is good (useful/signal) or bad (not useful).

For any subject, we have 28 ICs and hence 28 scale-maps, $SM_1, SM_2, \dots, SM_{28}$. The i^{th} scale-map, SM_i , is generated using the i^{th} column vector, \mathbf{x}_i , of the mixing A^{-1} , we use some experts (well experienced researchers) who can interpret / analyze EEG to label each scalp map (and hence each \mathbf{x}_i) as signal (useful component with label 1) or as artifact (not useful with label 0). Figure 3 displays the 28 scalp maps corresponding to a particular subject. In Fig. 3-1, we have labeled the useful components as good and the remaining ones as bad. For example, the 1st, 6th and 18th components are labeled as bad as these possibly correspond to eye blinking. We emphasize that there may be some scalp-maps on which experts may not agree about classifying the maps as artifacts or not.

In this way we generate a total of 980 column vectors, each representing an independent component. Each such component is labeled as good or bad. The generalization capability of any machine learning system designed from data heavily depends on variability in the training data. If the training data set has representation from the entire population, then the performance of system designed from such data is expected to be good. Thus, data from just one subject may not (usually will not) capture the desired level of variability even if such data are generated in different session of the same subject. In this investigation, we have used data from 35 subjects.

Although, we cannot claim that this data set will have the required level of variability, it is expected to do a reasonably good job.



We have summarized some useful scalp map in the table 3-1, it shows the scalp maps which has been chosen for brain activity researches. And the table 3-2 is the useless components which we want to remove it.

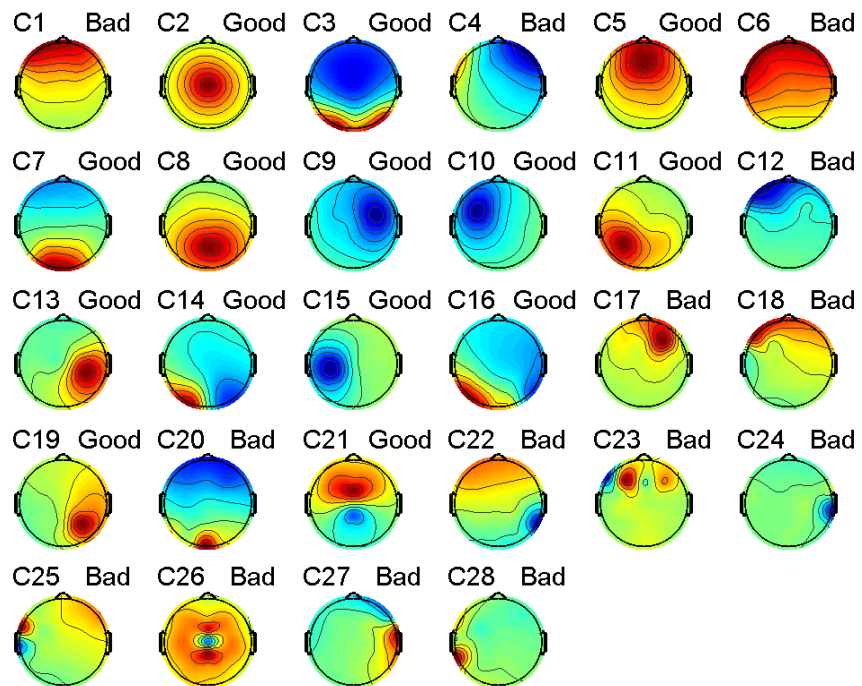


Figure 3-1. A set of illustrative ICA scalp map labeled by experts.



Table 3-1. An example of useful component set.


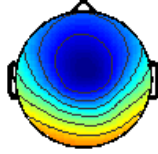
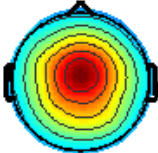
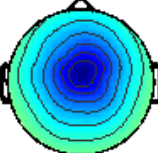
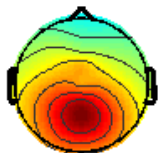





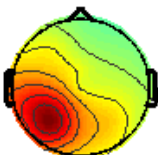
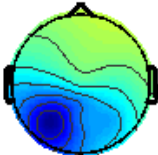



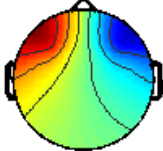


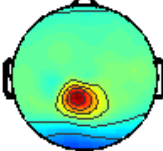

Component Name	Positive Scalp-Map	Negative Scalp-Map
Frontal		
Central		
Parietal		
Occipital		
Bi-lateral		
Left mu		
Right mu		

Table 3-2. A example set of useless components.

Component Name	Example 1	Example 2
Eye-Blinking		
Muscle Noise		
Others		



3-2. Supervised Methods

In this study, we use the supervised method. The reason has two concerns. First, the scalp-map has the same feature that we could predefine it before training. Second, the unsupervised method can't separate the useful / useless components. Cause there has no way to detect the super Gaussian distribution on the scalp map in different position, but the supervised method could solve these questions in a easily way. In this section, we introduce three basic and widely used classifiers, multilayer perceptron neural network (MLP), radial basis function neural network (RBF), and support vector machine (SVM) in our investigation.



3.2.1. Multilayer Perceptron Neural Network

We have used the back-propagation algorithm for training MLP networks with single hidden layer. The MLPTAN and MLPRBF both are multilayer perceptron networks (Figure 3-2).

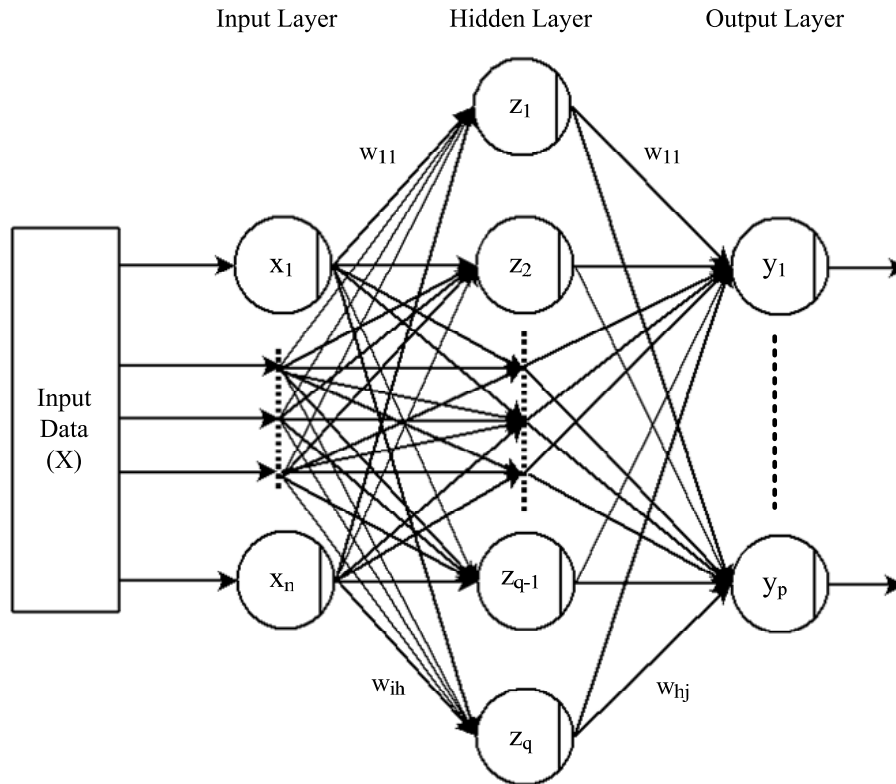


Figure 3-2. An example of one hidden layer neural network architecture.



Let us explain the architecture of MLP neural network. First, in a network, we want to perform the function:

$$f(X) = D \tag{1}$$

We have an input $X = (x_1, x_2, \dots, x_n)^T$, and perform the desire output $D = (d_1, d_2, \dots, d_p)^T$. Then, from the beginning of neural network, for the input layer:

$$S(x_i) = x_i, i = 1, \dots, n \tag{2}$$

$S(\cdot)$ is the sigmoid function of a neuron, depend on what learning scheme we want. Then, for the hidden layer:

$$z_h = \sum_{i=0}^n w_{ih} S(x_i), h = 1, \dots, q \quad (3)$$

w_{ih} is the weight value between neurons from hidden layer to output layer. For the output layer:

$$y_j = \sum_{i=0}^n w_{hj} S(z_h), j = 1, \dots, p \quad (4)$$

w_{hj} is the weight value between neurons from input layer to hidden layer. At the finish of neural network, $S(y_j)$ is the output of the neural network.

And we define the mean squared error for a neural network :

$$E = \frac{1}{2} \sum_{j=1}^p (d_j - S(y_j))^2 \quad (5)$$

In the back-propagation scheme, we can calculate the gradient using the chain rule of calculus between the hidden layer and output layer:

$$\frac{\partial E}{\partial w_{hj}} = \frac{\partial E}{\partial S(y_j)} \frac{\partial S(y_j)}{\partial y_j} \frac{\partial y_j}{\partial w_{hj}} \quad (6)$$

The back-propagation between input layer and hidden layer:

$$\frac{\partial E}{\partial w_{ih}} = \frac{\partial E}{\partial S(z_h)} \frac{\partial S(z_h)}{\partial z_h} \frac{\partial z_h}{\partial w_{ih}} \quad (7)$$

Then we could update the weight between the layers, for hidden to output layer weights:

$$w'_{hj} = w_{hj} + \Delta w_{hj} = w_{hj} + \eta \left(-\frac{\partial E}{\partial w_{hj}} \right) \quad (8)$$

η is the learning rate in the neural network. For input to hidden layer weights:

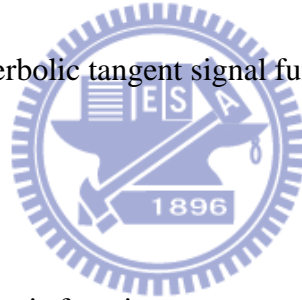
$$w'_{ih} = w_{ih} + \Delta w_{ih} = w_{ih} + \eta \left(-\frac{\partial E}{\partial w_{ih}} \right) \quad (9)$$

Both of MLPTAN and MLPRBF use the log-sigmoid transfer function at the output layer:

$$S(x) = \frac{1}{1 + e^{-x}} \quad (10)$$

The only difference between the two networks is in the signal function of the hidden layer. The MLPTAN uses hyperbolic tangent signal function:

$$S(x) = \frac{2}{(1 + e^{-2x})^{-1}} - 1 \quad (11)$$



The MLPRBF uses the radial basis function:

$$S(x) = e^{-x^2} \quad (12)$$

3.2.2. Radial Basis Function Neural Network

The RBF network (RBFNN) uses exactly three layers: input layer, basis function layer and output layer. Unlike MLP, the activation functions of the hidden nodes are not of sigmoid type, rather each hidden node represents a radial basis function. The

transformation from the input space to the hidden space is nonlinear but each node in the output layer computes just the weighted sum of the outputs of the previous layer, i.e., each output layer node makes a linear transformation. Thus the difference between MLPRBF (with just one hidden layer) and RBFNN lies not only in the signal function of the output layer, but also the learning scheme. The learning of RBF network is usually performed in two phases. An unsupervised learning method is applied to estimate the basis function parameters. Then a supervised learning method, such as gradient descent or least square error estimate, is applied to tune the network weights between the hidden layer and the output layer. For the detail of tuning the network weights, we need a training data set $T = \{X_k, d_k\}_{k=1}^Q$. And the function is generated by taking a weighted linear superposition of the basis functions:

$$f(X) = \sum_{k=1}^Q w_k \phi(\|X - X_k\|) \quad (13)$$

$\|X - X_k\|$ is the distance between the input $X = (x_1, x_2, \dots, x_n)^T$ and a training data point $X_i \in T$. From the requirement like (1), we have the equation:

$$\sum_{i=1}^Q w_i \phi(\|X - X_i\|) = d_k, \quad k = 1, \dots, Q \quad (14)$$

In order to recast this system of equations into matrix form, we introduce some definitions:

$$D = (d_1, d_2, \dots, d_Q)^T \quad (15)$$

$$W = (w_1, w_2, \dots, w_\varrho)^T \quad (16)$$

$$\Phi = \begin{pmatrix} \phi(\|X_1 - X_1\|) & \dots & \phi(\|X_1 - X_\varrho\|) \\ \vdots & \ddots & \vdots \\ \phi(\|X_\varrho - X_1\|) & \dots & \phi(\|X_\varrho - X_\varrho\|) \end{pmatrix} \quad (17)$$

Then we can get a equation from (X):

$$D = \Phi W$$

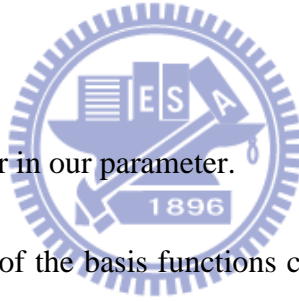
There, we can solve W from (X):

$$W = \Phi^{-1} D \quad (18)$$

And, our distance function is the Gaussian functions:

$$\phi(x) = e^{-\beta x^2} \quad (19)$$

β is the spread parameter in our parameter.



However, the parameters of the basis functions can also be tuned using gradient descent technique.

3.2.3. Support Vector Machine

The basic SVM [40-42] is formulated for a two class problem. If the training data are linearly separable, then SVM finds an optimum hyperplane that maximizes the margin of separation between the two classes.

Given a training set (X, Y) , $\mathbf{x}_i \in X$, $\mathbf{x}_i \in \mathfrak{R}^p$ and $y_i \in Y$, the class label associated with \mathbf{x}_i ; $y_i \in \{-1, +1\}$, the learning problem for SVMs is to find the weight vector \mathbf{w} and bias b such that they satisfy the constraints:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \text{ for } y_i = +1 \quad (20)$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \text{ for } y_i = -1 \quad (21)$$

and the weight vector \mathbf{w} minimizes the cost function

$$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w}. \quad (22)$$

The constraints written in (20)-(21) can be combined as

$$y_i (\mathbf{x}_i \cdot \mathbf{w} + b) \geq +1 \quad \forall i. \quad (23)$$

In our case, \mathbf{x}_i is a column vector of A and $+1$ and -1 can be views as good component and bad component. If the training points are not linearly separable, then there is no hyperplane that separates them into positive and negative classes. In this case, the problem is reformulated considering the slack variables $\xi_i \geq 0$; $i=1,2,\dots,N$.

For most \mathbf{x}_i , $\xi_i = 0$. The constraints are now modified as follows:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 - \xi_i \text{ for } y_i = +1 \quad (24)$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 + \xi_i \text{ for } y_i = -1 \quad (25)$$

$$\xi_i \geq 0, \quad \forall i. \quad (26)$$

The SVM then finds \mathbf{w} , minimizing

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N \xi_i \quad (27)$$

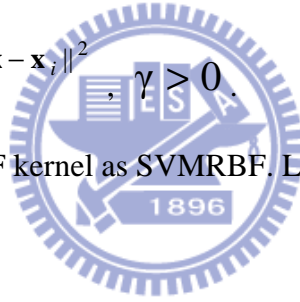
subject to constraints (24)-(26). The regularization parameter C controls the trade off between the complexity of the system and the number of misclassifications.

If the training points are not linearly separable, a nonlinear mapping φ is used to map the training data from \mathfrak{R}^p to some higher dimensional feature space H , with an expectation that the data may be linearly separable in H . The mapping is implicitly realized using a kernel function. There are different types of kernels that are popular.

Here we shall use only the RBF kernel:

$$K(\mathbf{x}, \mathbf{x}_i) = e^{-\gamma \|\mathbf{x} - \mathbf{x}_i\|^2}, \quad \gamma > 0. \quad (28)$$

We call SVM with the RBF kernel as SVMRBF. Like MLP and RBF network.



3-3. Training Algorithm Structure

Let us explain our design schemes using the network MLPTAN. The ICA components (i.e., the \mathbf{x}_i s) are partitioned randomly into 10 folds, each of equal (to the extent possible) size. We leave out one of the 10 folds for validation while the remaining 9 folds are used for training the system. To do this, the first issue to address is the choice of number of hidden layers and number of nodes in each hidden layer. In

all our experiments we use just one hidden layer. To find the best choice for the number of hidden nodes, we use a second (inner) level 10-fold cross validation as illustrated in Fig. 3-3. For MLPTAN we choose the best architecture by considering networks with number of neurons in the hidden layers in {10, 11, 12, 13, 14, 15}. Once we find the best structure, we take that architecture and trained the network using the outer level 9-folds. The trained network is then tested on fold that is left out. Then the entire process of random partitioning into 10-folds and computing the performance of the network using the optimal parameter found by a second level of cross-validation is repeated 5 times. The average error and its standard deviation over these 5 partitions are then reported. Similar experiments are then conducted with MLPRBF, RBFNN, and SVMRBF. For MLPTAN also we find the best architecture by considering networks with number of neurons in the hidden layers in {10, 11, 12, 13, 14, 15}. For the RBFNN network, two parameters are to be decided: the number of hidden nodes and the spread parameter of the Gaussian basis function. For this we have considered 35 choices : five choices for the number of hidden nodes, {60, 65, 70, 75, 80} and seven choices for the spread parameters, {1.5, 1.75, 2, 2.25, 2.5, 2.75, 3}. For the SVM classifier also we need to find suitable values for C and spread. To choose the optimal pair, we have considered 104 choices: eight choices for the

number of RBF scaling factor, {0.5, 1, 1.5, 2.5, 5, 10, 20, 25} and 13 choices for the box constraint value, {0.01, 0.1, 0.15, 0.2, 0.25, 0.3, 1, 1.5, 1.75, 2, 3, 4, 5}.

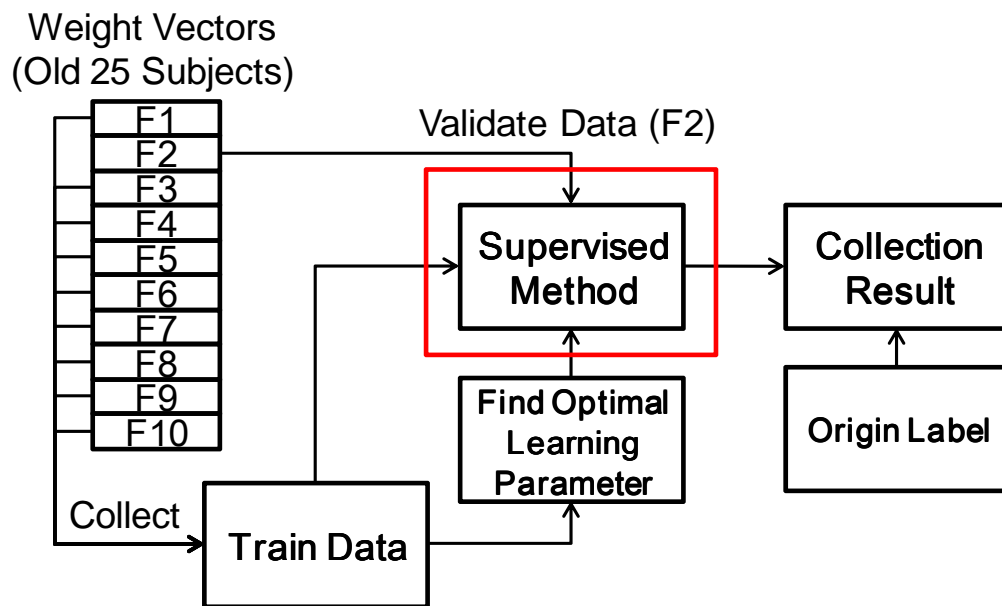


Figure 3-3. Ten-fold cross-validation of training algorithm structure.

3-4. Cross Validation

Cross-validation, sometimes called rotation estimation, is a method for evaluating the result performance of a classifier. It is mainly used in the whole labeled train data, and estimates how accurately a classifier will perform the prediction. Moreover, suppose we have a classifier with one or more unknown parameters, and a

data set to which the classifier can be fit the train data. The fitting process optimizes the classifier parameters to make the classification of train data as well as possible. In our experiment, we use the 10-fold cross-validation. The origin data set is partitioned into 10 subsets (folds). The folds are selected so that the sampling numbers is approximately equal in all the folds. In our classifier of a dichotomous classification, this means that each fold contains roughly the same proportions of the two types of class labels. The detail algorithm of 10-fold cross-validation is in 3-3. The advantage of this method over repeated random sub-sampling is that all observation we used for both training and validation, and each observation is used for validation exactly once. From the rounds of 10-fold, each fold perform a classifier once. Then we collect the classifiers from 10 rounds, and test the data set with these 10 optimal classifiers.

3-5. Testing Algorithm Structure

In the second set of experiments, we use a fusion type scheme which is designed to assess the effect of variability in subjects. Here we use a 10-fold cross validation on the data obtained from 25 training subjects (700 normalized weight vectors, 331 good and 369 bad). As done earlier, we leave out one of the 10 folds for validation while

the remaining 9 folds are used for training the system. In this experiment, in order to choose the optimal architecture, we do not use any inner level cross validation because we have new data on 10 subjects (280 normalized weight vectors, 83 good and 197 bad) for testing. We use the entire data from the 9-folds to find the best architecture for MLPTAN using the left out 10th fold as the validation set. As candidate architectures, we consider the same set of possible hidden nodes, {10, 11, 12, 13, 14, 15}, as done earlier.

Once we find the best structure (in terms of misclassifications on the validation set), we take that trained network for future use. For example, suppose the 10 folds are labeled as F_i ; $i=1, \dots, 10$. When we leave fold F_k out for validation, we get the classifier C_k , the best MLPTAN trained using F_i ; $i=1, \dots, 10$; $i \neq k$. In this way, we get 10 different classifiers C_k ; $k=1, \dots, 10$. Each of these classifiers is designed based on the training data from the 25 subjects.

Now each of these 10 classifiers is tested on the test data obtained from the 10 new subjects. Each classifier generates a value 0 or 1. For a given test data, \mathbf{x}_i , let y_k^i be output from classifier C_k . Now we compute a score S_i for the test data point \mathbf{x}_i

as $S_i = \sum_{k=1}^{10} y_k^i / 10$. In a typical fusion process $S_i \geq 5$ will be interpreted as good

component; otherwise, a bad component. However, all decisions made in this manner

may not necessarily be correct. So it might be better to use a threshold on S to make the decision. A decision with a higher threshold usually will indicate more confidence on the decision.

In the above discussion we have illustrated the process using MLPTAN. Similar experiments are also conducted with MLPRBF and RBFNN. To find the optimal parameters for MLPRBF, RBFNN, and SVMRBF we use the same set of candidate choices as done with the corresponding tool in the first set of experiments.

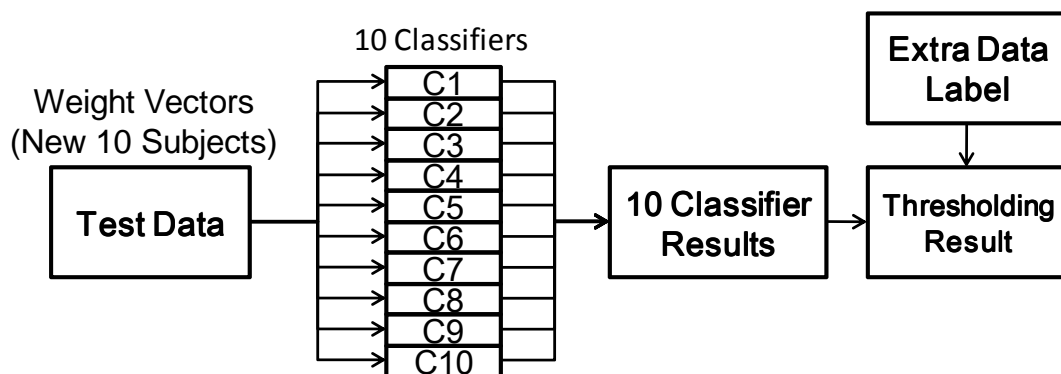


Figure 3-4. Testing algorithm through ten-fold cross validation training classifiers.

4. Experiment Results

4-1. Cross Validation Results of Training Data

Table 4-1 reports the average accuracy and standard deviation through four different machine learning models. We can clearly find that the SVMRBF performs the best accuracy about 92.6%. The second better performing model is RBFNN whose accuracy can reach about 90.6%. The other two neural networks, MLPTAN and MLPRBF perform similar accuracy. In addition, we can see that the standard deviations of these models are very low and consistent (not reach 1%). It means that the training models had been trained as a good structure with the optimal parameters. Therefore, the training experimental results are shown that it might be possible to have automatic systems for useful independent component selection. It is a strong indicator of the fact that with adequate training data it may be possible to design a “universal machine” for selection of good / useful independent components. Such systems would be extremely useful for real-time applications. The table 4-2 is the training parameters for performance evaluation from testing data.

Table 4-1. The accuracy of 10-fold cross validation.

Supervised Method	Accuracy Rate \pm Standard Deviation
MLPTAN	83.5% \pm 0.0082
MLPRBF	83.8% \pm 0.0133
RBFNN	90.6% \pm 0.0085
SVMRBF	92.6% \pm 0.0037

Table 4-2. The parameters of supervised methods.

Validate Fold	MLPTAN Neurons	MLPRBF Neurons	RBFNN Maximum Neurons	RBFNN Spread	SVMRBF Gamma	SVMRBF Box Constraint
Fold 1	11	12	70	3.0	2.0	2.5
Fold 2	12	10	80	2.5	2.0	5.0
Fold 3	10	13	80	3.0	2.25	5.0
Fold 4	13	13	75	3.0	2.5	1.5
Fold 5	12	13	70	2.75	2.0	1.0
Fold 6	15	14	60	2.5	2.0	1.0
Fold 7	10	10	70	2.75	2.25	1.0
Fold 8	12	15	75	2.75	2.5	2.5
Fold 9	13	10	80	3.0	2.0	0.5
Fold 10	10	11	65	2.75	2.5	1.0

4-2. Evaluation Performance of Testing Data

To evaluate the performance of the training structure from the four different machine learning models, we collect other different 10 subjects EEG datasets and applied ICA to extract the testing components. Each subject also has 28 components in each session. For real-life application, we individually test the accuracy performance of each subject. Figure 4-1 plots the average accuracy of testing datasets from 10 subjects via different threshold score. Different color dotted lines mean the average accuracy of testing datasets using different training structures with thresholds from 0.1 to 0.9. Short vertical lines on each threshold point mean the standard deviation of average accuracy from 10 subjects. According to the Figure 4-1 results, we have two major findings. The first finding is that the classification accuracy of four different models are increasing when the threshold is also increasing from 0.1 to 0.6 and then decreasing. The RBFNN model performs the best accuracy (92.5%) under threshold 0.6 and the local optimum threshold values 0.6 and 0.7 will lead to the better classification performance according to the performance curves. Moreover, we collect the classification accuracy of each subject under threshold 0.6 in Table 4-3. The numerator value in brackets means the number of correct classified components

and the denominator value means the totally 28 extracted components.

The other finding is that SVMRBF (the black line) performs the more stable classification performance. All classification accuracies of 10 different thresholds are over 85%. Therefore, if we cannot find the global optimum value of the threshold, we can use SVMRBF to be a general model of useful components selection. It can guarantee the classification performance is 85% at least. In other words, 85% accuracy means that there are four misclassified components in 28 components.

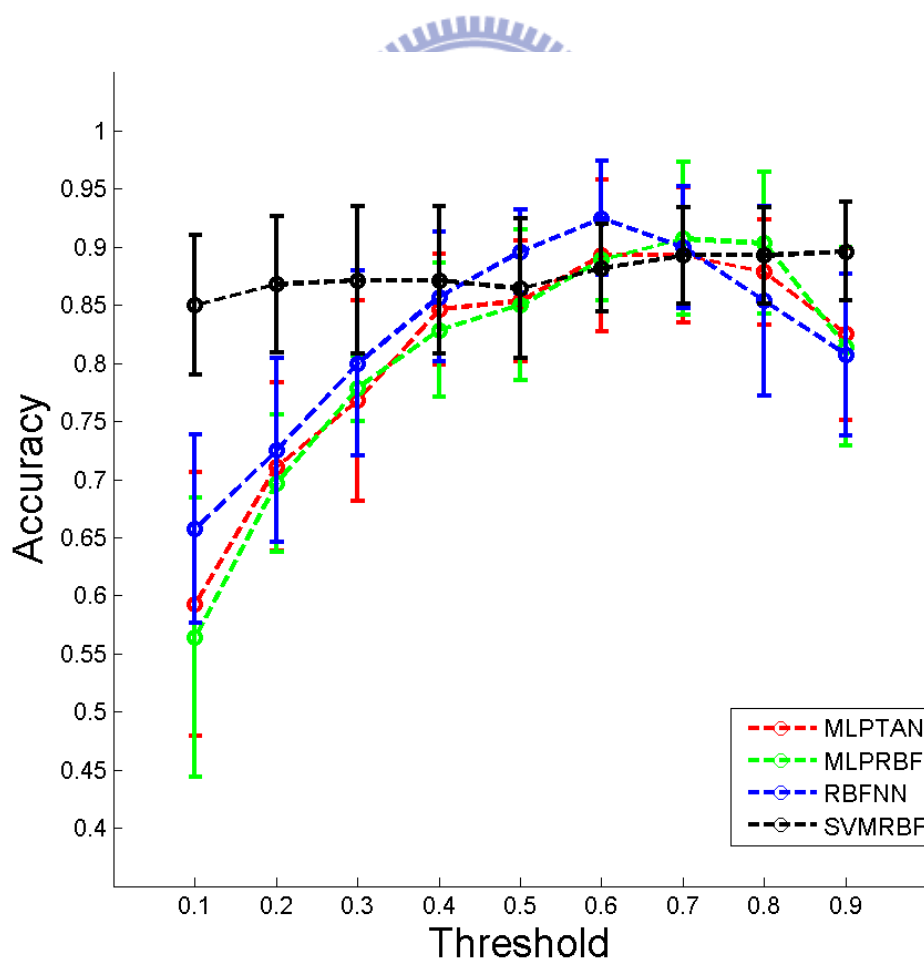


Figure 4-1. Average accuracy of testing datasets from 10 subjects via different threshold score.

Table 4-3. Classification accuracy of each subject under threshold 0.6.

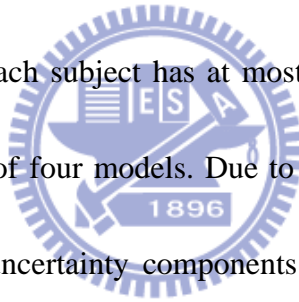
Subjects	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Average
MLPTAN	85.7% (24/28)	89.3% (25/28)	85.7% (24/28)	96.4% (27/28)	78.6% (22/28)	100% (28/28)	89.3% (25/28)	92.9% (26/28)	92.9% (26/28)	82.1% (23/28)	89.3% (250/280)
MLPRBF	82.1% (23/28)	89.3% (25/28)	85.7% (24/28)	92.9% (26/28)	85.7% (24/28)	89.3% (25/28)	89.3% (25/28)	89.3% (25/28)	92.9% (26/28)	92.9% (25/28)	88.9% (249/280)
RBFFNN	92.9% (26/28)	89.3% (25/28)	89.3% (25/28)	96.4% (27/28)	82.1% (23/28)	96.4% (27/28)	92.9% (26/28)	100% (28/28)	92.9% (26/28)	92.9% (26/28)	92.5% (259/280)
SVMRBF	82.1% (23/28)	92.9% (26/28)	89.3% (25/28)	89.3% (25/28)	85.7% (24/28)	92.9% (26/28)	82.1% (23/28)	89.3% (25/28)	89.3% (25/28)	89.3% (25/28)	88.2% (247/280)

For real-life application, we do not know the new observed component is useful or useless. In other words, that means we cannot know the desire output of each observed data. Therefore, we need to calculate the positive predictive value (*PPV*) from the confusion matrix to know the predictive performance. In this study, the *PPV* means the useful component predictive rate (*UCPR*). Equation (28) shows the *PPV* formula in which *TP* means true positive value and *FP* means false positive value.

$$PPV(UCPR) = \frac{TP(\text{True Useful Components})}{TP(\text{True Useful Components}) + FP(\text{False Useful Components})} \quad (28)$$

Corresponding to this study, *TP* means the number of true useful components

and FP means the number of false ones. In $UCPR$ equation, we have the assumption that the FP value is as small as possible because these false useful components are always from bad components (noise). Figure 4-2 plots the average $UCPR$ from 10 subjects via different thresholds. According to Figure 4-1 results, RBFNN performs the best accuracy performance when the threshold set at 0.6. Hence, we collect the $UCPR$ of each subject under threshold 0.6 in Table 4-4 and find that RBFNN still performs the best $UCPR$ performance (86%). In Table 4-4, the numerator value means the number of true useful components and the denominator value means true useful components add false ones. Each subject has at most three false useful components during the prediction process of four models. Due to the useful components are less than bad components, some uncertainty components from bad components will be easy to misclassify and affect the $UCPR$ significantly. Hence, we need to increase the population of useful components to enhance the stability and performance of the automatic scheme.



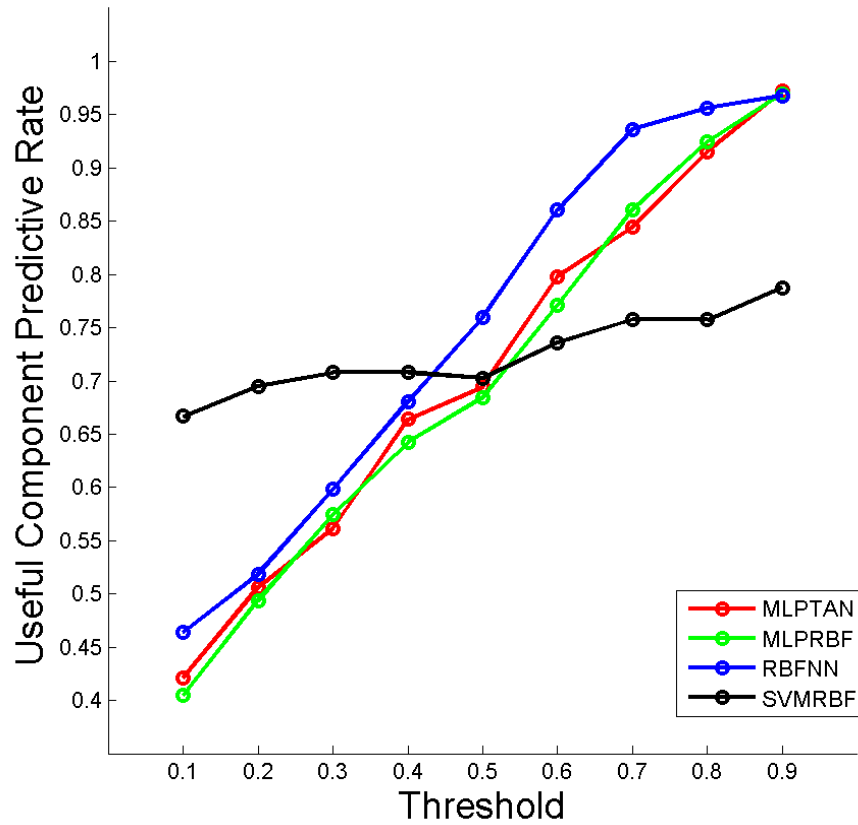


Figure 4-2. Average useful component predictive rate from 10 subjects via different threshold score.

Table 4-4. Useful component predictive rate of each subject under threshold 0.6.

Subjects	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Average
MLPTAN	6/9	8/10	8/11	4/5	6/9	8/8	6/8	8/9	12/12	5/8	71/89 (79.8%)
MLPRBF	6/10	9/12	8/11	4/6	8/11	7/9	6/8	8/10	12/12	6/7	74/96 (77.1%)
RBFNN	6/7	8/10	8/10	4/5	6/8	8/9	7/9	9/9	12/12	6/7	74/86 (86.0%)
SVMRBF	7/12	9/11	9/12	4/7	8/11	8/10	6/10	9/12	12/13	6/8	78/106 (73.6%)

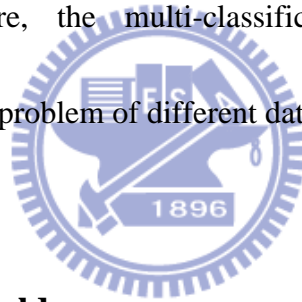
In Figure 4-2, the *UCPR* will be monotonically increasing in terms of the increasing thresholds. It shows the higher threshold performs the better predictive performance. In addition, although SVMRBF performs the best training performance, we can find it does not perform the best accuracy in testing evaluation process. The predictive performance of SVMRBF is also not stable for real-life application in comparison with other three models.

In summary, we suggest that RBFNN model will be the better model for the automatic scheme of useful component selection. It can also perform the better predictive performance for real-life application. Regarding the optimal threshold of RBFNN model, we suggest setting value at 0.6 is enough to have better performance. Since the number of good components is much smaller than the bad components, the training process may give more importance to the noise components. We plan to explore the utility of data replication as well as generation of additional data through rotation or negation. All these are part of future investigation.

5. Discussion

5-1. The Difference between Training and Testing

At Training session, the best accuracy was SVMRBF (92.6%), the second was RBFCC (90.4%). However, the testing session showed different results. The best accuracy was RBFNN (92.5%) in test session, and the SVMRBF accuracy (89.6%) was going to the third. This difference could introduce that the different dataset fitted different classifier. Therefore, the multi-classification method was a good consideration to eliminate this problem of different dataset.



5-2. The Optimal Threshold

In order to compare the different classifiers, we needed to choose the same threshold. Here we chose the threshold at 0.6. But the optimal threshold is different between classifiers. The RBFNN, MLPRBF, and MLPTAN had the local maximum accuracy in threshold 0.6 to 0.7. And the SVMRBF had the maximum accuracy at threshold 0.9. Thus, the result was a concern for implementing the multi-classification.

5-3. The Positive Predictive Rate (PPR)

In chapter 4, we introduced the PPR to find the best classifier from the ration between the true positive and good prediction outcome. The PPR showed the high accuracy of classifier, then the true positive would be high and the false negative would be low. The PPR results evidenced this phenomenon at threshold 0.6. However, there were some problems that if we reduced origin good label numbers, the PPR would be increase. We had two reasons to explain this problem. First, the labeling of scalp-map was defined by expert neuroscientist, and the good and bad label number was dependent on subjects. Second, the highest accuracy rate eliminated a problem which from the high threshold performed very low false positive, and if the true positive is exist, the PPR would be one. But we chose the highest accuracy rate rather than high threshold. Therefore, the PPR was a good factor to choose the better classifier.

6. Conclusion

Noise removal in EEG signal is an essential and important step in any application of EEG. If independent component analysis is used for this purpose, then selection of useful independent components is the most crucial step. Typically this selection is done by human experts who decide looking at the scalp maps. This manual intervention not only makes it dependent on the availability and subjectivity of experts, but also it becomes a stumbling block in BCI and in any other real-time application involving EEG. In this investigation, we have demonstrated that machine learning approaches can solve this problem to a great extent. In particular, we have used four machine learning tools. Although all of them are quite effective, the SVM with RBF kernel appears to be the most consistent and hence suitable one. Our investigation also indicates that the performance of a trained system on different subjects could be different, but this effect is not very severe and can be eliminated using fusion. Normally in a fusion scheme, decisions are made based on majority voting. In our investigation, using a simple score function we have found that higher vote does not necessarily mean better performance of a classifier system. Our

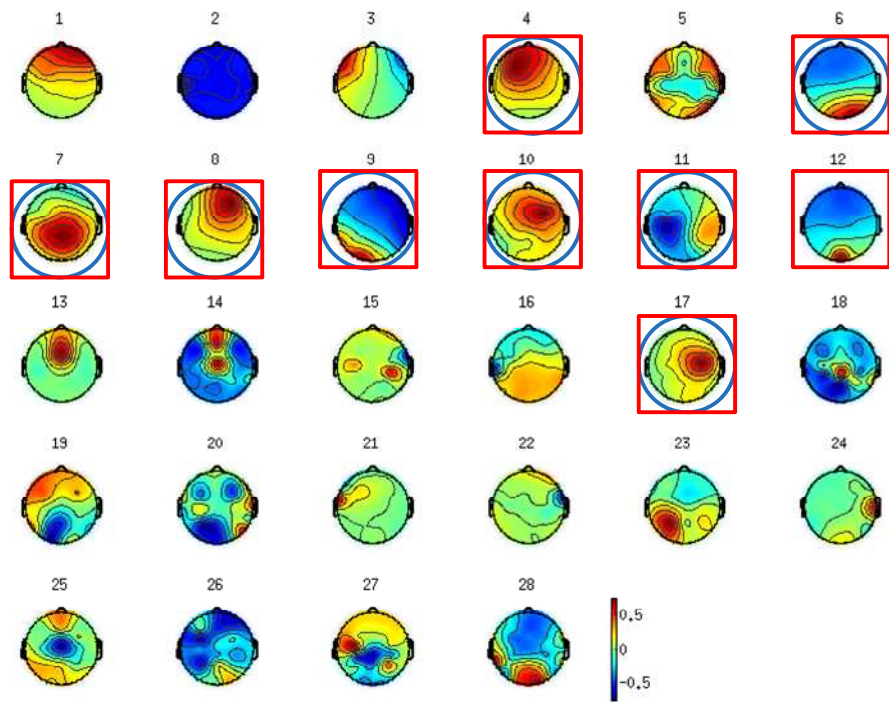
investigation indicates that it is possible to design a “universal system” for such job, given adequate training data.

Since the number of good components is much smaller than the bad components, the training process may give more importance to the poor components. We plan to explore the utility of data replication as well as generation of additional data through rotation or negation. All these are part of future investigation.

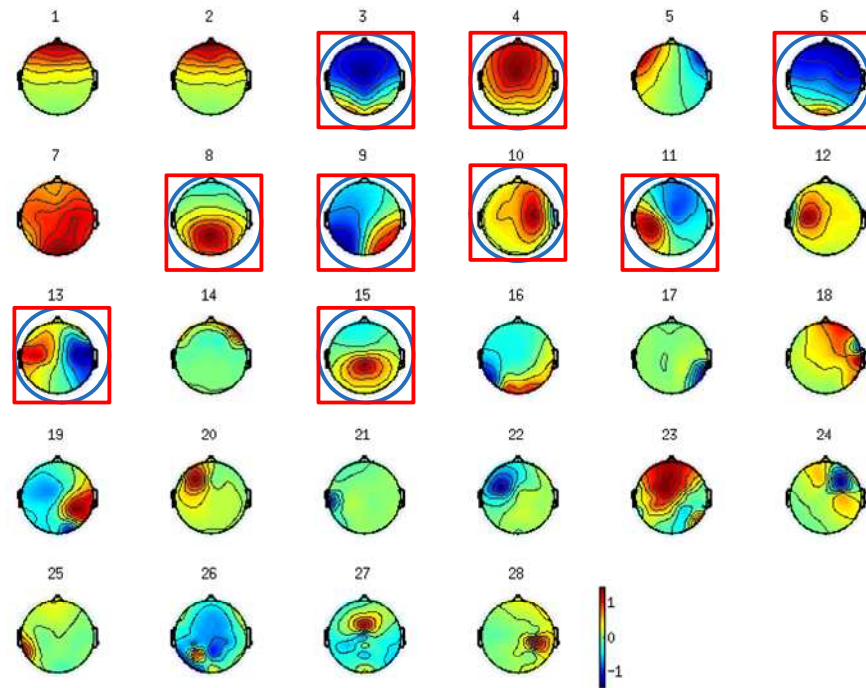


7. Future Works

In Fig. 7-1, the labeled useful component is marked with a blue circle, and the prediction of useful component is marked with red rectangle. It presents that the component which only has red rectangle or blue circle is the misclassification. In order to make the clustering correctly, the component which only has both red rectangle and blue circle is considered. However, it makes a problem between the components in Fig. 7-1 (a) no. 7, and (b) no. 8 and 15. These components are useful component (it is parietal) by manual selection, but the classifier selects the (a) no. 7 and (b) no. 8 as useful component without the (b) no.15. Because in the training data, there are some components which have high weight in an electrode position on scalp map, the similar component is in the Fig. 7-1 (b) no. 27 (this is an example in the frontal region). Therefore, if we want to implement the component clustering, we need to improve the automatic component selection more effective, or enhance the dimension of the weight matrix to extract more detail features.



(a).



(b)

Figure 7-1. An illustration of how a scalp map is drawn.

7-1. The Enhancement of Feature

The enhancement of weight matrix is a good choice of future experiment. Not only we can reduce the misclassification of insufficient data dimensions, but also the new matrix has the possibility that the component clustering could remove the bad component by itself. In our experiment, we used 28-channel recording. However, there exist 32, 64, or 128 channels to record the EEG signal. It is a big trouble if we change the recording channel, the classifier of automatic selection needs to be trained again for new input data. Therefore, we could consider the scalp-map as our new feature for training classifiers. The different channel recording would generate the same scalp map topology, which is the matrix that we called color map. The color map could be a large or small matrix, which is selected by the concentration of points on the scalp map. We can train and evaluate classifiers by the color maps as input data. The component clustering system could be performed by using the color maps, which supplies the more detail from scalp-map for future experiment. These hypothesis need to design a new system scheme to implement, that is another topics in future works.

7-2. Auto Clustering System

The component clustering system is a designing topic to extent the ability of independent component analysis. In recent researches, the ICA can separate the EEG signals into independent components and draw the scalp map by weight matrix. And we can analyze specific independent component with corresponding scalp-map. Thus, we could point out the EEG activities from different regions. Like the component based on occipital lobe has the advantage on drowsiness analysis, the component based on prefrontal cortex usually concentrates the eye-blinking affect. We could base on these characteristic phenomena to understand the association between the brain activities and behaviors. If we could achieve this system, for the engineers or psychologists which has no neuroscience background is an important tool to introduce them into brain analysis. Currently, mostly brain researches based on ICA have lots of conclusion between component activities and behaviors. Therefore, we could implement the component clustering system for these researches in brain computer interface or real-time application. This work will changes the application of raw EEG based analysis system in the world.

Reference

- [1] M. Iwasaki, C. Kellinghaus, A. Alexopoulos, R. Burgess, A. Kumar, Y. Han, H. Luders, and R. Leigh, "Effects of eyelid closure, blinks, and eye movements on the electroencephalogram," *Clinical Neurophysiology*, vol. 116, pp. 878-885, 2005.
- [2] I. Goncharova, D. McFarland, T. Vaughan, and J. Wolpaw, "EMG contamination of EEG: spectral and topographical characteristics," *Clinical Neurophysiology*, vol. 114, pp. 1580-1593, 2003.
- [3] D. Brunner, R. Vasko, C. Detka, J. Monahan, C. Reynolds III, and D. Kupfer, "Muscle artifacts in the sleep EEG: Automated detection and effect on all-night EEG power spectra," *Journal of Sleep Research*, vol. 5, pp. 155-164, 1996.
- [4] D. Mantini, M. Perrucci, S. Cugini, A. Ferretti, G. Romani, and C. Del Gratta, "Complete artifact removal for EEG recorded during continuous fMRI using independent component analysis," *Neuroimage*, vol. 34, pp. 598-607, 2007.
- [5] P. Allen, G. Polizzi, K. Krakow, D. Fish, and L. Lemieux, "Identification of EEG Events in the MR Scanner: The Problem of Pulse Artifact and a Method for Its Subtraction," *Neuroimage*, vol. 8, pp. 229-239, 1998.
- [6] P. Allen, O. Josephs, and R. Turner, "A Method for Removing Imaging Artifact from Continuous EEG Recorded during Functional MRI," *Neuroimage*, vol. 12, pp. 230-239, 2000.
- [7] A. Erfanian and B. Mahmoudi, "Real-time ocular artifact suppression using recurrent neural network for electro-encephalogram based brain-computer

- interface," *Medical and Biological Engineering and Computing*, vol. 43, pp. 296-305, 2005.
- [8] P. He, M. Kahle, G. Wilson, and C. Russell, "Removal of Ocular Artifacts from EEG: A Comparison of Adaptive Filtering Method and Regression Method Using Simulated Data," in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, 2005, pp. 1110-1113.
- [9] P. He, G. Wilson, and C. Russell, "Removal of ocular artifacts from electro-encephalogram by adaptive filtering," *Medical and Biological Engineering and Computing*, vol. 42, pp. 407-412, 2004.
- [10] P. He, G. Wilson, C. Russell, and M. Gerschutz, "Removal of ocular artifacts from the EEG: a comparison between time-domain regression method and adaptive filtering method using simulated data," *Medical and Biological Engineering and Computing*, vol. 45, pp. 495-503, 2007.
- [11] A. Soong and Z. Koles, "Principal-component localization of the sources of the background EEG," *IEEE Transactions on Biomedical Engineering*, vol. 42, pp. 59-67, 1995.
- [12] Z. Koles, J. Lind, and A. Soong, "Spatio-temporal decomposition of the EEG: a general approach to the isolation and localization of sources," *Electroencephalography and Clinical Neurophysiology*, vol. 95, pp. 219-230, 1995.
- [13] T.-P. Jung, S. Makeig, C. Humphries, T. Lee, M. McKeown, V. Iragui, and T. Sejnowski, "Removing electroencephalographic artifacts by blind source separation," *Psychophysiology*, vol. 37, pp. 163-178, 2000.

- [14] T.-P. Jung, C. Humphries, T. Lee, S. Makeig, M. McKeown, V. Iragui, and T. Sejnowski, "Removing electroencephalographic artifacts: comparison between ICA and PCA," in *Neural Networks for Signal Processing VIII, 1998. Proceedings of the 1998 IEEE Signal Processing Society Workshop*, 1998, pp. 63-72.
- [15] S. Makeig, A. Bell, T. Jung, and T. Sejnowski, "Independent Component Analysis of Electroencephalographic Data," *Advances in Neural Information Processing Systems*, pp. 145-151, 1996.
- [16] A. Flexer, H. Bauer, J. Pripfl, and G. Dorffner, "Using ICA for removal of ocular artifacts in EEG recorded from blind subjects," *Neural Networks*, vol. 18, pp. 998-1005, 2005.
- [17] C. Joyce, I. Gorodnitsky, and M. Kutas, "Automatic removal of eye movement and blink artifacts from EEG data using blind component separation," *Psychophysiology*, vol. 41, pp. 313-325, 2004.
- [18] T.-P. Jung, C. Humphries, T. Lee, S. Makeig, M. McKeown, V. Iragui, and T. Sejnowski, "Extended ICA Removes Artifacts from Electroencephalographic Recordings," *Advances in Neural Information Processing Systems*, pp. 894-900, 1998.
- [19] P. LeVan, E. Urrestarazu, and J. Gotman, "A system for automatic artifact removal in ictal scalp EEG based on independent component analysis and Bayesian classification," *Clinical Neurophysiology*, vol. 117, pp. 912-927, 2006.
- [20] Y. Li, Z. Ma, and W. Lu, "Automatic removal of the eye blink artifact from EEG using an ICA-based template matching approach," *Physiological Measurement*, vol. 27, p. 425, 2006.

- [21] S. Romero, M. A. Mananas, S. Clos, S. Gimenez, and M. J. Barbanøj, "Reduction of EEG artifacts by ICA in different sleep stages," in *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, 2003, pp. 2675-2678 Vol.3.
- [22] R. Vigario, "Extraction of ocular artefacts from EEG using independent component analysis," *Electroencephalography and Clinical Neurophysiology*, vol. 103, pp. 395-404, 1997.
- [23] A. Bell and T. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural computation*, vol. 7, pp. 1129-1159, 1995.
- [24] T. Lee, M. Girolami, and T. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources," *Neural computation*, vol. 11, pp. 417-441, 1999.
- [25] F. De Martino, F. Gentile, F. Esposito, M. Balsi, F. Di Salle, R. Goebel, and E. Formisano, "Classification of fMRI independent components using IC-fingerprints and support vector machine classifiers," *Neuroimage*, vol. 34, pp. 177-194, 2007.
- [26] T. Kochiyama, T. Morita, T. Okada, Y. Yonekura, M. Matsumura, and N. Sadato, "Removing the effects of task-related motion using independent-component analysis," *Neuroimage*, vol. 25, pp. 802-814, 2005.
- [27] M. McKeown, S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski, "Analysis of fMRI data by blind separation into independent spatial components," *Human Brain Mapping*, vol. 6, pp. 160-188, 1998.

- [28] S. Fitzgibbon, D. Powers, K. Pope, and C. Clark, "Removal of EEG Noise and Artifact Using Blind Source Separation," *Journal of Clinical Neurophysiology*, vol. 24, p. 232, 2007.
- [29] F. La Foresta, N. Mammone, and F. Morabito, "PCA-ICA for automatic identification of critical events in continuous coma-EEG monitoring," *Biomedical Signal Processing and Control*, 2009.
- [30] F. La Foresta, F. C. Morabito, B. Azzerboni, and M. Ipsale, "PCA and ICA for the extraction of EEG components in cerebral death assessment," in *Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on*, 2005, pp. 2532-2537 vol. 4.
- [31] C. Bugli and P. Lambert, "Comparison between principal component analysis and independent component analysis in electroencephalograms modelling," *Biometrical Journal*, vol. 49, pp. 312-327, 2007.
- [32] T.-P. Jung, S. Makeig, M. Westerfield, J. Townsend, E. Courchesne, and T. Sejnowski, "Analysis and visualization of single-trial event-related potentials," *Human Brain Mapping*, vol. 14, pp. 166-185, 2001.
- [33] H.-S. Huang, N. R. Pal, L.-W. Ko, and C.-T. Lin, "Automatic Identification of Useful Independent Components with a View to Removing Artifacts from EEG Signal," in *Neural Networks, 2009., 2009 IJCNN International Joint Conference on*, Atlanta, USA, 2009, p. 5.
- [34] C.-T. Lin, Y.-C. Chen, R.-C. Wu, S.-F. Liang, and T.-Y. Huang, "Assessment of driver's driving performance and alertness using EEG-based fuzzy neural networks," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, 2005, pp. 152-155.

- [35] C.-T. Lin, L.-W. Ko, I.-F. Chung, T.-Y. Huang, Y.-C. Chen, T.-P. Jung, and S.-F. Liang, "Adaptive EEG-Based Alertness Estimation System by Using ICA-Based Fuzzy Neural Networks," *IEEE Transactions on Circuits and Systems Part 1 Regular Papers*, vol. 53, p. 2469, 2006.
- [36] C.-T. Lin, R.-C. Wu, S.-F. Liang, W.-H. Chao, Y.-J. Chen, and T.-P. Jung, "EEG-based drowsiness estimation for safety driving using independent component analysis," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 52, pp. 2726-2738, 2005.
- [37] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, pp. 9-21, 2004.
- [38] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. New York: John Wiley & Sons, 2001.
- [39] T. Lee, *Independent component analysis: theory and applications*: Kluwer Academic Publishers Hingham, MA, USA, 1998.
- [40] V. Vapnik, *The nature of statistical learning theory*: springer, 2000.
- [41] S. Kumar, *Neural Networks: a classroom approach*: Tata McGraw-Hill, 2004.
- [42] S. Haykin, *Neural networks: a comprehensive foundation*: Prentice Hall, 2008.

Appendix

The confusion matrix from evaluation results:

(a). MLPTAN

T0.1	C1	C2	T0.2	C1	C2	T0.3	C1	C2	T0.4	C1	C2
C1	83	114	C1	82	80	C1	82	64	C1	81	41
C2	0	83	C2	1	117	C2	1	133	C2	2	156
0.59 (166/280)			0.71 (199/280)			0.77 (215/280)			0.85 (237/280)		
T0.5	C1	C2	T0.6	C1	C2	T0.7	C1	C2	T0.8	C1	C2
C1	75	33	C1	71	18	C1	65	12	C1	54	5
C2	8	164	C2	12	179	C2	18	185	C2	29	192
0.85 (239/280)			0.89 (250/280)			0.89 (250/280)			0.88 (246/280)		
T0.9	C1	C2	T1.0	C1	C2	C1: Good Component Class					
C1	35	1	C1	x	x	C2: Bad Component Class					
C2	48	196	C2	x	x	TX.X: Threshold X.X					
0.82 (231/280)			X								

(b). MLPRBF

T0.1	C1	C2	T0.2	C1	C2	T0.3	C1	C2	T0.4	C1	C2
C1	83	122	C1	82	84	C1	81	60	C1	79	44
C2	0	75	C2	1	113	C2	2	137	C2	4	153
0.56 (158/280)			0.7 (195/280)			0.78 (218/280)			0.83 (232/280)		
T0.5	C1	C2	T0.6	C1	C2	T0.7	C1	C2	T0.8	C1	C2
C1	76	35	C1	74	22	C1	68	11	C1	61	5
C2	7	162	C2	9	175	C2	15	186	C2	22	192
0.85 (238/280)			0.89 (249/280)			0.91 (254/280)			0.9 (253/280)		
T0.9	C1	C2	T1.0	C1	C2	C1: Good Component Class					
C1	32	1	C1	x	x	C2: Bad Component Class					
C2	51	196	C2	x	x	TX.X: Threshold X.X					
0.81 (228/280)			X								

(c). RBFNN

T0.1	C1	C2	T0.2	C1	C2	T0.3	C1	C2	T0.4	C1	C2
C1	83	96	C1	83	77	C1	82	55	C1	81	38
C2	0	101	C2	0	120	C2	1	142	C2	2	159
0.66 (184/280)			0.72 (203/280)			0.8 (224/280)			0.86 (240/280)		
T0.5	C1	C2	T0.6	C1	C2	T0.7	C1	C2	T0.8	C1	C2
C1	79	25	C1	74	12	C1	59	4	C1	44	2
C2	4	172	C2	9	185	C2	24	193	C2	39	195
0.9 (251/280)			0.93 (259/280)			0.9 (252/280)			0.85 (239/280)		
T0.9	C1	C2	T1.0	C1	C2	C1: Good Component Class					
C1	30	1	C1	x	x	C2: Bad Component Class					
C2	53	196	C2	x	x	TX.X: Threshold X.X					
0.81 (226/280)			X								

(d). SVMRBF



T0.1	C1	C2	T0.2	C1	C2	T0.3	C1	C2	T0.4	C1	C2
C1	82	41	C1	82	36	C1	80	33	C1	80	33
C2	1	156	C2	1	161	C2	3	164	C2	3	164
0.85 (238/280)			0.87 (243/280)			0.87 (244/280)			0.87 (244/280)		
T0.5	C1	C2	T0.6	C1	C2	T0.7	C1	C2	T0.8	C1	C2
C1	78	33	C1	78	28	C1	78	25	C1	74	20
C2	5	164	C2	5	169	C2	5	172	C2	9	177
0.86 (242/280)			0.88 (247/280)			0.89 (250/280)			0.90 (251/280)		
T0.9	C1	C2	T1.0	C1	C2	C1: Good Component Class					
C1	74	20	C1	x	x	C2: Bad Component Class					
C2	9	177	C2	x	x	TX.X: Threshold X.X					
0.90 (251/280)			X								