Studies on Multilayer Neural Networks Applied to Frequency Selective Signal Classification

STUDIES ON MULTILAYER NEURAL NETWORKS APPLIED TO FREQUENCY SELECTIVE SIGNAL CLASSIFICATION

Kazuyuki HARA

January 16, 1997

博　士　論　文

STUDIES ON MULTILAYER NEURAL NETWORKS APPLIED TO FREQUENCY SELECTIVE SIGNAL CLASSIFICATION

金沢大学 大学院 自然科学研究科
システム科学 専攻
電子・情報システム 講座

学　籍　番　号	92 - 2213
氏　　　　名	原　一之
主任指導教官名	中山　謙二

ACKNOWLEGEMENTS

Foremost, I would like to express my profound gratitude to Professor Kenji Nakayma for his helpful guidance throughout this work. I wish to thank Professor Kohki Matsuura, Professor Ken-ichiro Muramoto , Professor Tetsuro Funada, Professor Kiyoshi Nishikawa and the late Professor Ken-ichi Hayashi of Kanazawa University, and Professor Tsuyoshi Takebe of Kanazawa Institute of Technology for providing valuable comments concerning this study.

Secondly, I want to express my thank to Ms. Zhiqiang Ma, Mr. Kazushi Ikeda, Mr. Youhua Wang, Mr. Hiroshi Katayama and Mr. Kazuo Arai for useful discussions and kind helps.

I am grateful to my colleagues Mr. Yoshihiro Tomikawa and Mr. Seiji Miyoshi their various helps and comments on my study.

I am grateful also to colleague Mr. Ashraf A. Khalaf for his kind helps and interesting discussions.

I am grateful to Mr. Asadual Huq, Mr. Atse Yapi, Miss Gao Li, Mr. Kunihiko Imai, Mr. Yoshinori Kimura, Mr. Takuo Kato, Mr. Katsuaki Nishimura, Mr. Kazuhiro Minamimoto, Mr. Atsushi Ohsaki, Mr. Katsuyoshi Ohnishi, Mr. Hideki Kobori and all the member of Nakayama Laboratory.

I want to show my grateful to all the staff of Toyama Polytechnic College. I am grateful to former president Mr. Fusao Mikami and present president Mr. Shizuo Tada for their understanding of my study at Kanazawa University. I also want to express my grateful to Mr. Tadashi Watanabe who introduced me the neural networks. I am grateful to my colleagues Mr. Tadashi Watanabe, Mr. Minoru Takata, Miss Kaoru Karaki, Mr. Hitoshi Akiyama, Mr. Akio Hirano and Mr. Naoyuki Sato for their thoughtful helps and assistance.

Last, I am most thankful to my wife and daughter, Ritsuko and Tomoni Hara, for their constant encouragement in many difficult and strange hours. I am also grateful to my parents Yoshiyuki and Fumiko Hara for growing in me interest in science.

Contents

Chapter 1

INTRODUCTION

This thesis scopes a multi-layer neural network (MLNN), and analyze its mechanism of the pattern classification. In this chapter, first, the history of study of the MLNN is introduced, then the back ground of this study is explained, and finally, the organization of this thesis is shown.

1.1 History of Study of Multilayer Neural Networks

In 1969, M. L. Minsky and S. A. Paper [1] demonstrated some limitation of a single-layered neural network whose activation function is a linear function by applying many classification problems. This is called the linear neural network in this paper. The single-layer neural network consists of the input layer and the output layer.

They showed that the neural network performs the linear mapping from the input into the output, however, nonlinear mapping cannot be solved by this type of neural network. The exclusive OR problem was one of the famous examples that could not solve by the linear neural networks. After that, the neural network no longer used as the all-purpose classifier, and the studies of the neural network were declined.

In 1986, D. E. Rumelhart, J. L. McClelland and the PDP research group wrote a book named Parallel Distributed Processing[2]. Its subtitle is explorations in the microstructure of cognition. The aim of this book was building a theory of cognition in the microstructure, however, some of them can be applied to the machine learning for the pattern classification, optimization and so on. The back-propagation (BP) algorithm and the Boltzmann Machine are introduced in this book. Especially, BP algorithm applied to the MLNN using the sigmoid activation function, is useful for the pattern classification. The BP algorithm is characterized by followings: (1) BP algorithm is based on Least Mean Square (LMS) algorithm developed by Widrow and Hoff [3]. The LMS algorithm is an important member of the family of stochastic gradian-based algorithms[4]. (2) Sigmoid function is a monotonically increasing nonlinear function, so nonlinear mapping of the input and output pattern can be realized. (3) Moreover, the hidden layer, which locates between the input layer and the output layer, is added. Since using a nonlinear activation function, by increasing the number of the hidden layers, the classification performance can be increase[5]. In [2], the algorithm is given and many examples are shown, however, theoretical analysis and designing methods are not sufficiently discussed.

After PDP is published, many researchers and engineers applied the BP algorithm to many applications, and they showed usefulness of the MLNN using BP algorithm. However, there are many rule of thumb to train the network but it is still difficult to use the MLNN using BP algorithm for real problems.

Theoretically, Funahashi[6] proved that the two-layered NN can approximate any continous function with any accuracy if a large number of hidden units are used. Amari gave some mathematical foundations of neruocomputing by using information theory[7], and Widrow and Lehr [8] reviewed several neural network models and gave some insight to the models. Levin, Tishby and Solla proposed a statistical approach for the MLNN [9]. T.Poggio[10] proposed a model especially for the approximation of functions based on the regularization theory. This model can be realized by using network called the radial basis function network. This network is different from the MLNN discussed in this thesis.

On the design point of view, Wada and Kawato introduced an information criterion using corss validation and applied it to decide the number of hidden units for approximtely correct (PAC) learning [11]. Vapnik-Chervonenkis (VC) dimension[12] is also used to reducing the number of the hidden units. There are many papers related to reducing the number of hidden units and fast convergence[13, 14, 15, 16, 17].

As mentioned above, there are many papers related to specific problems, however, theoretical analysis of superiority of the MLNN against convensional methods and its mechanizm that realize the superiority is not well discussed.

1.2 Background

Recently, neural networks (NNs) have been applied to the signal processing fields, including signal detection [18, 19, 20, 21, 22], digital demodulation [23, 24, 25, 26], digital signal classification [27, 28, 29]. In these applications, the NN methods can provide good performances. Furthermore, there are many papers comparing multilayer NNs (MLNNs) and statistical methods in the application point of view. For example, pattern classification performance, complexity of structure for implementation and computations have been taken into account in comparison in Tsoi and Atlas[30, 31] , Gish[32], and Lippmann [5], respectively. From these results, the MLNN method has recognized to be superior to linear Signal Processing (LSP) methods under some conditions. However, these conditions have not been well discussed from theoretical point of view.

In this thesis, comparison between the MLNN and the LSP methods used in signal classification is discussed. Usually, the MLNN method is useful for arbitrary pattern classification. On the other hand, the LSP method is good for detecting the signals specified by frequency components. The purpose of this thesis is to investigate usefulness of the MLNN method in the signal processing field, therefore, the signals specified by frequency are considered. Thus, the signals are classified based on their frequency components.

Furthermore, the observation period is very short. This means that the number of the signal samples is set to be very small. Since, in this case, frequency information

may be lost to some extent, the signal classification becomes more difficult. This kind of limitations appears in the digital communication, the signal processing, and the real time image processing [32] fields. From practical view point, computational complexity is also limited. Namely, the comparison will be discussed based on length of the signal sequence and complexity of implementation.

Since the MLNN is a non-parametric model, the generalization for untrained data is an important criterion. Furthermore, robustness for noisy signal classification is also compared. Through theoretical and experimental results, we derive the conditions, under which we can estimate which method is useful in frequency selective signal classification.

1.3 Organization of the Thesis

In chapter 2, the pattern classification mechanism of the MLNN is analyzed. The classification realized by the MLNN can be seen as dividing the input pattern space and form the class region by hyper-plane formed by the connection weights. Then, the degree of freedom to form the class region is analysed[33].

In chapter 3, two training data selection methods are proposed to guarantee generalization[34] . For the MLNN, to select the training data to guarantee the generalization is important. I pointed out that the important data for the purpose is to be near the class boundary or the connection weights, and to select the data near the class boundary, two algorithms of the pairing method and the pairing and training method are proposed. Computer simulation is carried out to investigate usefulness of the two methods.

In chapter 4, an application for frequency selective classification is investigated through computer simulations. The computer simulation is carried out under two conditions that are uniform activation function[33] and several activation functions[35, 36]. In the case of uniform activation function, the sigmoid function is used. In the case of several activation functions, the sigmoid function, the sinusoidal function and the gaussian function are used. For the classification task, multi-frequency signal is used.

In chapter 5, an application for frequency selective classification by using the linear signal processing (LSP) methods are discussed[33, 37]. The analysis is carried out based on the pattern classification rather than the frequency analysis. Given a signal of N samples, it can be viewed as N dimensional vector. Then, dividing the N-dimensional space to assign the same class signals to the same class region. Therefore, the classification performance is analyzed by the degree of freedom to form the class region in the N-dimensional space.

In chapter 6, by computer simulation, the classification performance of the MLNNs and the LSP methods are compared based on classification accuracy, the number of samples of the signal and the computation. Moreover, the dial-tone signal, which is the concrete signal of the multi-frequency signal, is used. The dial-tone signal is used for push button phone. From the results, the MLNN can achieve high classification performance with small computations compared with the LSP methods

for complex problems[33, 37, 38].

Chapter 7 summarizes and concludes some results. The reason of superiority of the MLNN methods against the LSP methods is due to high degree of freedom to form the class region of the MLNN, and the superiority is clear when the computations of the algorithm or the number of the samples of the signals are limited. When the computations or the number of the samples of the signal is not limited, the classification performance of the MLNN methods and the LPs methods are almost the same.

Chapter 2

PATTERN CLASSIFICATION BY MULTILAYER NEURAL NETWORKS

2.1 Introduction

In this chapter, a mechanism of the classification by the multilayer neural network (MLNN) is analyzed. For this purpose, the classification by the MLNN is treated as division of the input pattern space to match the pattern classes.

First, the structure of the MLNN is introduced. Next, the function of the hidden layer and the output layer is shown. Then the performance of the classification of the MLNN is analyzed based on the degree of freedom to form a class region. From this analysis, the MLNN has a high degree of freedom to choose the combination of the connection weights from the hidden layer to the output layer. Realization of to the correct classification is shown. The ability of training is also discussed.

2.2 Pattern Classification by Neural Networks

2.2.1 Structure of Multilayer Neural Network

The multilayer neural network used in this paper is a two-layered neural network consists of the input layer, one hidden layer and the output layer. The discussion in this chapter is based on this type of the MLNN. However, the results can be applied to the MLNN more than one hidden layer, then the generalization of the discussion will not be lost. Fig.2.1 shows an example of the two-layer MLNN with three input units, three hidden units and three output units.

Assuming that the network consists of N input units, J hidden units and K output units. The length of N and K is set to be as the same as the number of samples of the signal and the number of the signal classes. Accordingly, N samples signal is represented as an N-dimensional vector. The target signal is set for one class that one output unit is activated and the others are inhibited. In this chapter, there are $p = 1 \sim P$ signal classes, and there are $m = 1 \sim M$ of N-dimensional signals. The mth signal of pth class is denoted $\boldsymbol{x}_{pm} = \{x_{pm}(n), n = n_0 \sim n_0 + N - 1\}$. Here, n_0 is the starting point of the observation. Each sample of the N-samples signal is applied to the input unit in parallel, so Nth unit received $x_{pm}(n_0 + N - 1 - n)$.

The input potential of the jth hidden unit is denoted by net_j. This is calculated as a weighted sum of the input-hidden unit connection weight w_{nj} multiplied by the input signal as follow.

$$net_j = \sum_{n=0}^{N-1} w_{nj} x_{pm}(n_0 + N - 1 - n) + \theta_j \tag{2.1}$$

Here, θ_j is the bias of the jth hidden unit. The output of this unit is calculated by using the hidden unit's activation function f_H.

$$y_j = f_H(net_j) \tag{2.2}$$

In the output layer, the input and the output of the unit is calculated by the same manner. The input potential of the kth output unit is denoted net_k and the output of this unit is y_k. Then if the connection weight from the jth hidden to the kth output unit is w_{jk} and the activation function of the output unit is $f_O(\cdot)$,

$$net_k = \sum_{j=0}^{J-1} w_{jk} y_j + \theta_k \tag{2.3}$$

$$y_k = f_O(net_k) \tag{2.4}$$

The training algorithm of the connection weights is the supervised training. In this paper, Back-propagation algorithm is used. Therefore, the activation function used as $f_H(\cdot)$ and $f_O(\cdot)$ should be a differentiable function.

2.2.2 Pattern Classification

In the MLNN, the input signal applied to the input layer is semi-classified in the hidden layer, and is classified to suitable class in output layer. In this section, I show what kind of classification is done by above process. Moreover, the degree of freedom of the pattern classification is discussed in Sec.2.3.

The MLNN performs a kind of mapping which map the input patterns to discrete classes. This mapping is achieved by linear combination described by Eq.(2.1) and Eq.(2.3), and the nonlinear mapping of Eq.(2.2). So, first, the pattern classification property of single neuron[39] is shown.

The input pattern is $\boldsymbol{x}_{pm} = \{x_{pm}(n), n = 0 \sim N-1\}$, the connection weights are denoted $\boldsymbol{w} = \{w_n, n = 0 \sim N-1\}$, and the bias is θ. To classify the input patterns into two classes of $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, $\boldsymbol{w}$ and θ which satisfy the following Eq.(2.5) must be exist.

$$\left.\begin{aligned} \boldsymbol{x}_{pm} \in \boldsymbol{X}_1,\ \textstyle\sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta > 0 \\ \boldsymbol{x}_{pm} \in \boldsymbol{X}_2,\ \textstyle\sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta < 0 \end{aligned}\right\} \tag{2.5}$$

Here, a monotonically increasing function includes non-continuos function is assumed as the non-linear function correspond to Eq.(2.2) and (2.4). 0 threshold linear activation function shown in Eq.(2.6) is an example. y is the output unit output.

$$y = \begin{cases} 1, & \sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta > 0 \\ 0, & \sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta < 0 \end{cases} \tag{2.6}$$

The right side inequality of Eq.(2.5) shows that the N-dimensional space is divided into two regions by one hyper-plane of next equation.

$$\sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta = 0. \tag{2.7}$$

In other words, if there are the connection weight $\boldsymbol{w}$ and the bias θ satisfy Eq.(2.6), then the region includes the patterns belong to $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ divided into two regions by the hyper-plane. This sort of the signal set is said to be linear separable.

The activation function of the MLNN described in Sec.2.2.1 is a differentiable function. The sigmoid function written as Eq.(2.8) is an example of the function.

$$f(net) = \frac{1}{1 + e^{-net}} \tag{2.8}$$

$f(net)$ is a monotonically increasing function and the output is an analog value in the range of [0,1], however, as Eq.(2.9), using an output threshold of 0.5, Eq.(2.5) work as the division condition of the classes.

$$y \begin{cases} > 0.5, & \sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta > 0 \\ < 0.5, & \sum_{n=0}^{N-1} w_n x_{pm}(n) + \theta < 0 \end{cases} \tag{2.9}$$

The signal classification by single hidden unit is as the same as one by single neuron. Then, all the patterns are divided into two classes at the output of the hidden unit. In general, the pattern classification is not always linearly separable, so accuracy of the classification at the hidden layer is not guaranteed. When there are J hidden units, all the patterns are classified into 2^J subclasses at the hidden layer. These subclasses are formed by the connection weights from the input layer and the hidden layer.

On the other hand, to match the input of Eq.(2.5) to the output of the hidden units, and to match the output of the equation to the single unit of the output layer, it can be found that all patterns are classified into two classes by the single output unit. In other words, the subclasses formed by the hidden layer is combined into two regions by single output unit. If the subclasses are linearly separable at the hidden layer, it is possible to classify the signals correctly. The degree of freedom of the classification by the MLNN is discussed in Sec. 2.3.

2.3 Analysis of Degree of Freedom for Pattern Classification

Analysis of the degree of freedom of pattern classification is done based on the degree of freedom to form the class region at the output layer.

In this section, to analyze the degree of freedom of the pattern classification, the number of combinations of the subclasses to be linearly separable is counted out. The subclasses are formed at the hidden layer. If the subclasses are linearly separable, they are classified correctly by the hyper-planes formed by the connection

weights from the hidden layer to the output layer. Therefore, to count the number of the combinations of subclasses linearly separable at the hidden layer is equal to estimate the degree of freedom to form the regions in the N-dimensional space to achieve the classification.

T. Cover counted the number of dichotomies of the random patterns by single neuron in the statistical sense[39, 40]. A. Kowalczyk extended Cover's result to the fist hidden layer[41]. J. Makhoul showed partitioning capabilities of two-layer neural networks[42]. However, in this paper, the analysis is curried out in the deterministic sense.

For analysis, assuming that the activation function of the hidden unit is a threshold function. This assumption is as the same as in Sec.2.2.1. By using the threshold function substituting for the sigmoid function, the class region will be slightly changed, however, the number of the hyper-plane is kept as the same as the one using the sigmoid function. The region can be adjusted through the training process, and if the number of the hyper-plane is the same, the degree of freedom to form the class region will be the same. To ease the discussion here, the linear threshold function is used in the followings. Two classes classification of two dimensional input patterns by the MLNN that consists of two input units and one output unit is considered.

In the followings, analysis is curried out for two cases: two hidden units, three hidden units.

(1) Case of using two hidden units.

From Eq.(2.5), all the patterns $\boldsymbol{X}$ are divided into two sub-regions by single hidden unit. In the case of two hidden units, the input space is divided into four subregions as depicted in 2.2(a) by the connection weights from the input layer to the hidden layer. In this figure, two solid lines are the hyper-planes formed by the connection weights.

The four subregions are combined into two class regions by single output unit, then two class regions formed by the combination of these four subregions should be linearly separable. In general, the number of the hidden units is J, the MLNN has a degree of freedom to form the class regions formed by linearly separable combination of the subregions out of 2^J combinations. In Fig.2.2(a), linearly non-separable combinations of the subregions are I and III form a region for one class, and II and IV form a region for the other and its opposite combination[43]. The combination except above can combine the subregions into two class regions. To ease the following analysis, the hidden unit output space is used.

Since the activation function of the hidden unit is the threshold function, the output of the hidden unit will be 1 or 0. So, combinations of the output of the hidden unit are $(H_1, H_2) = \{(0,0),(0,1),(1,0),(1,1)\}$. Here, the first hidden unit is denoted by H_1 and the second hidden unit is denoted by H_2, respectively. As depicted in Fig.2.2(b), these four patterns can be considered as vertices of unit a square in R^2. Length of the side is unity. The class boundary is the tilted line in the figure. Then, counting the number of combinations of the subregions to be linearly separable by the output unit comes down to solving the number of the hyper-plane linearly separating the vertices of the square.

Since the bias is included into the connection weights, the hyper-plane can be shifted from the center of the square. So, at the beginning, the case of using the bias is analyzed. Two, three and four vertices' separations are considered, because, some input patterns will be classified by using a partial pattern. Total of the vertices are two, the number of combination to select two vertices from four, and to select one vertex for one class from two vertices, then ${}_4C_2 \times {}_2C_1 = 12$ is the possible combinations of linearly separable. Here, ${}_mC_n$ denote the number of combinations of n objects that can be made from a set of m objects. ${}_mC_n$ is calculated as ${}_mC_n = m!/(n! \times (m-n)!)$. In the case of total of the vertices is three, ${}_4C_3 \times \{{}_3C_1 + {}_3C_2\} = 24$ is a possible number of linearly separable combinations. All the vertices are used. ${}_4C_4 \times \{{}_4C_1 + ({}_4C_2 - 2) + {}_4C_3\} = 12$ is the number of the combinations. In the case of each two vertices is separated, as described in Fig.2.2(a), the relation of two vertices is in exclusive OR, these two vertices are linearly non-separable. Then these two combinations are excluded from the number of the combinations.

The counting of $12 + 24 + 12 = 48$ is the number of combinations of linearly separable. The ratio of all the combinations ($48+2$(the number of the combinations of linearly non-separable)) to the number of combinations of linearly separable are $48/50 = 0.96$, so it can be concluded that a degree of freedom of forming a class region of the MLNN is high.

(2) Three hidden units case

In the case of three hidden units, the hidden unit outputs are represented as cube. That is $(H_1,H_2,H_3)=\{(0,0,0),(0,0,1),(0,1,0),(0,1,1),(1,0,0),(1,0,1),(1,1,0),(1,1,1)\}$. Figure 2.3 shows the hidden unit output space and states as vertices.

In this case, linear separability is considered based on vertices on the Three-dimensional plane and on the two-dimensional planes. Three-dimensional plane is consisted with four vertices of (1,0,0),(1,0,1),(0,1,0) and (0,1,1), as an example. (1,1,0),(1,1,1),(0,1,1) and (0,1,0) are vertices on two-dimensional plane.

There are same number of linearly separable and linearly non-separable combinations on a two-dimensional plane as two hidden units case. There are six two-dimensional planes in the cube, so from the result of two hidden unit case, the combination of vertices on two-dimensional planes are obtained.

For vertices on the three-dimensional planes, there are four combinations of vertices in exclusive OR. If two combinations of the vertices are in exclusive OR, they are linearly non-separable. The vertices {(0,0,0), (1,1,1)}, {(0,0,1), (1,1,0)}, {(0,1,0), (1,0,1)} and {(0,1,1), (1,0,0)} are in exclusive OR. Figure 2.3 shows one of the exclusive combination. White circles and black circles are linearly non-separable. To count the number of linearly non-separable combinations, there are three ways to select linearly non-separable combinations of the vertices, two, three and four. In the case of two combinations of vertices in exclusive OR, there are ${}_4C_2 \times \sum_i^{16-4} {}_{12}C_i$. For three combinations of vertices in exclusive OR, there are ${}_4C_3 \times \sum_{i=0}^{16-6} {}_{10}C_i$. For all combinations of vertices in exclusive OR, there are ${}_4C_4 \times \sum_{i=0}^{16-8} {}_8C_i$. Summation of above is 288. In this case, the number of all the combinations of the vertices is $\sum_{i=2}^{16} {}_{16}C_i \, (\sum_{j=1}^{16-i} {}_iC_j) = 5060$. So, ratio of linearly separable combination is (5060-480)/5060=0.905.

To generalize above discussion, the number of combinations of the vertices in exclusive OR on the N-dimensional hyper-plane included in the N-dimensional hyper-cube must be obtained. Here, the N-dimensional hyper-plane in the N-dimensional hyper-cube includes the vertices that are in exclusive OR. The coordinate of one vertex in exclusive OR on the N-dimensional hyper-plane in the N-dimensional hyper-cube is given by inverting each element of the coordinate of the other vertices. And the hamming distance of the vertices in exclusive OR on the N-dimensional hyper-plane in the N-dimensional hyper-cube is N. Then the number of combinations of the vertices in exclusive OR on the hyper-plane in N-dimensional hyper-cube is half of all the combinations of the vertices or $2^N/2 = 2^{N-1}$. The number of all combinations of the vertices in an N-dimensional hyper-cube is calculated by the following equation.

$$\sum_{j=2}^{2^N} {}_{2^N}C_j \Big(\sum_{m=1}^{j-1} {}_jC_m \Big) \tag{2.10}$$

So, by subtracting the number of the combinations that is linearly non-separable from Eq.(2.10), the number of the combinations of linearly separable in the N-dimensional hyper-cube can be solved.

The number of the combinations that include the vertices in exclusive OR on N-dimensional hyper-plane in the N-dimensional hyper-cube is given by

$$\sum_{i=2}^{2^{N-1}-1} {}_{2^{N-1}}C_i \Big(\sum_{k=1}^{i-1} {}_iC_k \Big(\sum_{j=0}^{2^N-2i} {}_{2^N-2i}C_j \Big) \Big). \tag{2.11}$$

Now, the number of the combinations of linearly non-separable is counted. In

4-dimensional hyper-cube, the number of the vertices is V, the number of the sides is E, the number of the faces is F, and the number of cells is C have some relations as follows[44].

$$V - E + F - C = 0 \tag{2.12}$$

This is called as Euler-Poincare's multi-cell body theorem. For hyper-cube, each value is as follows.

$$V = 16, E = 32, F = 24, C = 8$$

Assuming this theorem is true more than 5-dimensional hyper-cube, we can get the next table.

Table 2.1: Relation of vertices and cell of hyper-cube over 3-dimensional hyper-cell

dimension	vertex	cell(face for 3-dimension)
3	8	6
4	16	8
5	32	10
n	2^n	$2n$ $(n > 3)$

For N-dimensional hyper-cube, the number of linearly non-separable combination of the vertices on the N-dimensional hyper-plane is given by Eq.(2.11). The number of the linearly non-separable combination of the vertices on single $N-1$-dimensional hyper-plane is also given by Eq.(2.11), and is multiplied by the number

of cells. So, the number of the linearly non-separable combinations of the vertices is sum up from the number of the combination of the vertices on the N-dimensional hyper-plane to the number of the combination of the vertices on the 2-dimensional plane. The number of cells included in the N-dimensional hyper-cube is induced from Table2.1.

For example, in the case of 3-dimensional cube, the number of linearly non-separable combination of the vertices on 3-dimensional hyper-plane is given by next calculation.

$$\begin{aligned}
&\sum_{i=2}^{2^{3-1}-1} {}_{2^{3-1}}C_i\Big(\sum_{k=1}^{i-1} {}_iC_k\Big(\sum_{j=0}^{2^3-2i} {}_{2^3-2i}C_j\Big)\Big) \\
&= \sum_{i=2}^{3} {}_4C_i\Big(\sum_{k=1}^{i-1} {}_iC_k\Big(\sum_{j=0}^{8-2i} {}_{8-2i}C_j\Big)\Big) \\
&= 288
\end{aligned} \tag{2.13}$$

The number of linearly non-separable combination of the vertices on single 2-dimensional plane is given by Eq.(2.11) and the number of faces is given by Table2.1 as 6, then,

$${}_2C_2 \times {}_2C_1 \times \sum_{j=0}^{4} {}_4C_j \times 6 = 32 \times 6 = 192 \tag{2.14}$$

is the number of linearly non-separable combinations of the vertices on the plane. Then, the solution is given by sum up these results as $288 + 192 = 480$.

The number of combinations of all the vertices of the cube is given by Eq.(2.10) and is 5060, then the ratio of the number of combination of vertices of linearly separable is $(5060-480)/5060 = 0.905$. This result is showing the degree of freedom of the classification by three hidden units, then the ratio is decreased compare to the case of two hidden units, however, the number of the combinations of the vertices that are linearly separable is drastically increased.

On the other hand, if the biases for the output units are not used, the hyper-plane cannot be shifted. Then the vertices located at diagonal Cannot be classified into the same class. Therefore, the number of combination of the vertices that are linearly separable will be decreased. For two vertices separation with two hidden units, the number of the combination of vertices that are linearly separable is ${}_4C_2 \times {}_2C_1 = 12$, three vertices case is ${}_4C_3 \times \{({}_3C_1 - 1) + ({}_3C_2 - 1)\} = 16$, all the vertices case is ${}_4C_4 \times \{{}_4C_2 - 2\} = 4$. Then totally, the number of the combinations is 32. The ratio of all number of the combination of the vertices to the number of the combination of the vertices that are linearly separable is $32/50 = 0.64$ so, the degree of freedom to the classification is decreased. In general, the MLNN uses the bias for the output units, then former results of $48/50 = 0.96$ can be expected, and higher degree of freedom to the classification can be held.

When the sigmoid function is used as the activation function, above results can be changed. In this case, the hidden layer outputs are distributed near the vertices as shown as gray area in Fig.2.4.

If the distribution of the hidden unit outputs can be divided by the line hence, the results of the linear threshold function can be applied. r denotes some limit of distribution that the above results can be applied. Assume that the distribution of

the hidden unit outputs is the same, r is given by the circle whose radius is r and its tangential line, $r \fallingdotseq 0.42$. This value is approximately equal to the limit of the distribution from each vertex of 0.5. And after train the network, the hidden unit output tends to be 1 or 0 [45], so the distribution will be small. Or assuming that the distribution is small, the pattern classification performance of the MLNN will not be decreased. Therefore, the degree of freedom to form a class region using the sigmoid function is the same as that of using the linear threshold function.

From above analysis, the degree of freedom of the MLNN to form a class region or the number of the hyper-plane (kind of the connection weight from the hidden layer to the output layer) is high, so in the case of the input patterns are distributed widely and complicatedly, the MLNN can form the class regions. The reasons of this capability of forming a class region come from non-linearity of the activation function and the architecture of using hidden layer.

However, the MLNN is a non-parametric method and at the same time, it is trained by using relation of training patterns and its class [2], then convergence of training the network is not guaranteed. And the class region is decided by training patterns, so if the distribution of the training patterns is biased, the classification performance will be decreased. Therefore, the performance of the MLNN is depending on selection of the training pattern and the training method.

2.4 Learning Ability and Convergence Property

Supervised learning algorithms, like the back-propagation (BP) algorithm, were proposed to train the MLNN[2]. The supervised learning is used to train the MLNN. Thus, discussions on learning ability and convergence property are important.

As described in Sec. 2.2.2, for the MLNN, the classification problem is equivalent to dividing the N-dimensional space into several sub-spaces.

As mentioned before, the number of the signal samples is assumed to be small. This is further divided into the following two cases, (1) a very small number, and (2) a relatively small number. Furthermore, the circuit complexity, which is mainly determined by the number of the hidden units, is practically important. Two cases, (a) a small number of the hidden units, and (b) a large number of them, are taken into account.

In the case (1), the frequency components become vague. In other words, the regions, in which the signals of each class are distributed, are changed from their original distribution. Sometimes, the class regions are mixed and overlapped. However, if they are not overlapped, it is possible to separate the areas into the different classes.

In the case (2), the signals include accurate frequency components, and they are distributed in some specific regions. The regions of the different classes are separated. However, the boundary between them may be complicated and narrow. In the linear filter methods, the filter design is equivalent to approximate this boundary

by using several sets of the filter coefficients or the impulse response samples.

In the MLNN method, in order to achieve complete separation, which is to form the complicated boundary, many hidden units are required. For this reason, the learning converges slowly, and it is easily to be trapped into the local minimum. Therefore, the initial connection weights should be carefully selected.

On the other hand, if a small number of the hidden units are used, the complete separation is impossible. However, relatively high classification rate can be obtained due to high degree of freedom of forming the boundary as mentioned in Sec.2. In this case, stable and fast convergence can be obtained.

2.5 Summary

In this chapter, the classification using the MLNN is treated as division of the input pattern space to match the pattern classes. The degree of freedom of the MLNN to form the class region is analyzed and the number of the combination of vertices that are linearlity separable is counted for two and three hidden units, respectively. Moreover, the suggestion for expansion of the results for any number of the hidden units is given. From the results, the MLNN has high degree of freedom to form the class region. The reason of this result comes from non-linearly of the activation function and the architecture of having hidden layer.

The MLNN needs the training process, then the ability of training is discussed. If the number of the network parameters is more than one required to solve a problem, so adjustment of many parameters to obtain a solution will be difficult, then convergence speed becomes slow. On the other hand, if the number of the parameters is smaller, then, adjustment of small number of the parameter to obtain a solution is easy, however, their remains some residual error.

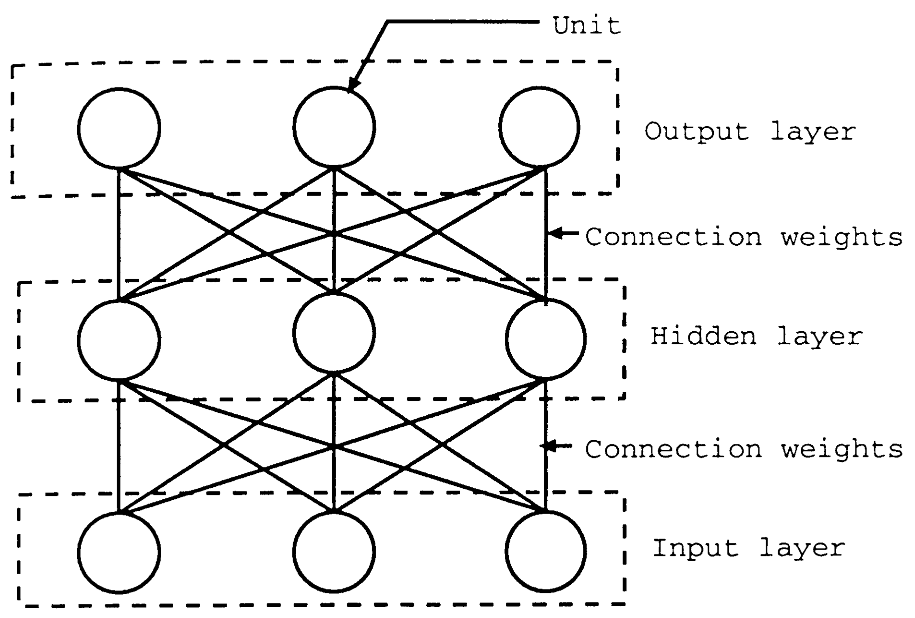


Figure 2.1: Example of architecture of multilayer neural network

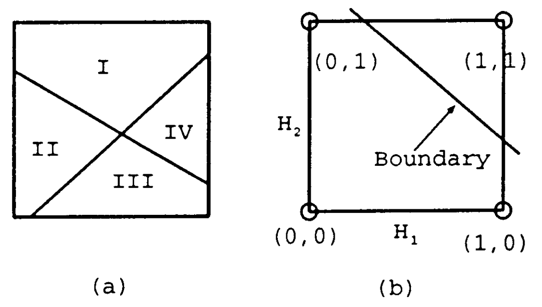


Figure 2.2: Signal detection region of MLNN with two hidden units. (a) Sub-class regions in input space. (b) Class boundary in hidden unit output space.

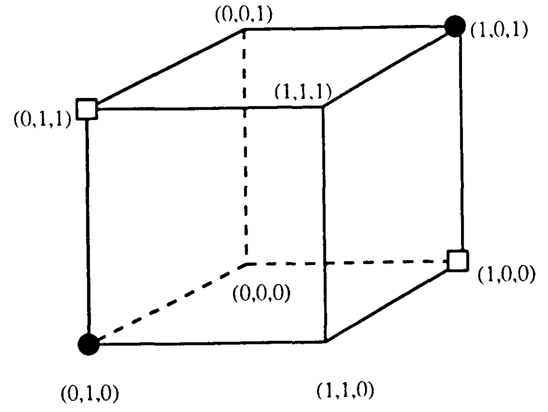


Figure 2.3: Hidden unit output space of three hidden units.

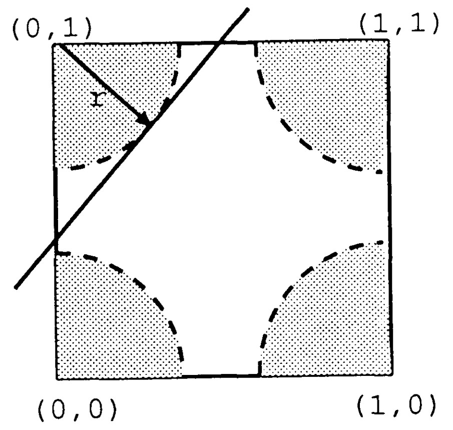


Figure 2.4: Distribution of hidden unit outputs with Sigmoid function.

Chapter 3

MINIMUM TRAINING DATA SELECTION FOR MULTILAYER NEURAL NETWORKS

3.1 Introduction

In the classification problems, a multilayer neural network (MLNN) trained by supervised learning algorithms are capable of extracting common features or rules of training data through a training process. This is a benefit of using the MLNN for the classification. However, the suitable architecture of the MLNN and a small number of training data are required. The error back-propagation(BP)[2] algorithm is a popular algorithm for solving the classification problems.

One of the main interests of the supervised learning algorithms is how to select the training data. A huge number of the training data may guarantee generality of the MLNN. On the other hand, it will require a very long training time. Therefore, it is desirable to reduce the number of the training data while maintaining generalization. Cachin [46] proposed the error-dependent repetition(EDR). Presentation probability of the training data is proportional to the MLNN output error. However, the entire data are used in the training process. M. Kutsuwada proposed iterate learning method to fix the generalization area[47]. This is one of the approaches to guarantee the generalization performance.

In this chapter, we propose a method to select the efficient training data, with which generalization is guaranteed[34]. The selected data can locate around the boundary between classes. This method can be applied to reduce in data memory and computations of off-line training, where a sufficient number of training data can be obtained in advance. Furthermore, it will be useful for an on-line training, where all training data cannot obtain at the beginning, rather they are gradually increased.

Efficiency of the proposed method is investigated through computer simulations. The BP algorithm is used to train the MLNN. Two kinds of problems are employed as examples.

3.2 Activation Functions

In this chapter, a two-layer MLNN is used to classify the data. N samples of a piece of data $\boldsymbol{x} = \{x(i), i = 1 \sim N\}$ is applied to the input layer. The ith input unit receives $x(i)$. The connection weight from the ith input to the jth hidden unit is denoted w_{ij}. The input potential net_j and the output y_j of the jth hidden unit are given by

$$net_j = \sum_{i=1}^{N} w_{ij}x(i) + \theta_j \tag{3.1}$$

$$y_j = f_H(net_j), j = 1 \sim J \tag{3.2}$$

$$f_H(net_j) = \frac{1 - e^{-net_j}}{1 + e^{-net_j}} \tag{3.3}$$

where, $f_H(\cdot)$ is an activation function in the hidden layer and θ_j is a bias. The input potential net_k and the output y_k of the kth output unit are given by

$$net_k = \sum_{j=1}^{J} w_{jk}y_j + \theta_k \tag{3.4}$$

$$y_k = f_O(net_k), k = 1 \sim P \tag{3.5}$$

$$f_O(net_k) = \frac{1}{1 + e^{-net_k}} \tag{3.6}$$

where $f_O(\cdot)$ is an activation function in the output layer.

The number of output units is equal to that of the classes. The MLNN is trained so that a single output unit responds to one of the classes.

3.3 Geometrical Property of Input and Output

Input of the jth hidden unit is expressed by Eq.(3.1). The input space can be separated into two regions by a line formed by $net_j = 0$ at the input of the hidden unit. A distance between this line and the input data is given by

$$d_j = \frac{\left|\sum_{n=1}^{N} w_{ij}x(i) + \theta_j\right|}{||\boldsymbol{w}_j||} = \frac{|\boldsymbol{net}_j|}{||\boldsymbol{w}_j||}, \tag{3.7}$$

$$\boldsymbol{w}_j = \{w_{ij}, i = 1 \sim N\}. \tag{3.8}$$

$||\boldsymbol{w}_j||$ is an L_2 norm of the weight vector $\boldsymbol{w}_j$. Then the input potential net_j is proportional to the distance d_j. The activation function Eq.(3.3) is a continuous monotonically increasing function, then the hidden unit output y_j is also continuous monotonically increasing with respect to the distance d_j. However, y_j is not a linear function of the distance.

The output of the output unit y_k is separated by the regions of $y_k > 0.5$ and $y_k < 0.5$. The input potential $net_k = 0$ provides a decision boundary. This is called a network boundary in this paper. The class boundary means the boundary of the input data classes. If the training converges, the network boundary will agree with the class boundary. Then a distance from the class boundary to a data is related to $|y_k - 0.5|$. In this case, the input potential of the output unit net_k is also related to the distance.

In conclusion, $|y_k - 0.5|$ and $|net_k|$ are continuous functions with respect to the distance between the data boundary and the input data.

3.4 Pairing Method for Training Data Selection

The proposed data selection method combines a training process and a pairing method. In this section, a pairing method is first described.

In this thesis, two classes $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ are taken into account for convenience. However, the proposed method can be applied to more than two classes.

In the pairing process, the nearest data of different classes evaluated using the Euclidean distance is selected. Let $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ be set of two data classes, and $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be element of them. $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are paired with each other through the following steps.

Step 1: Select $\boldsymbol{x}_1$ (or $\boldsymbol{x}_2$) from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$) randomly.

Step 2: Select $\boldsymbol{x}_2^p$ (or $\boldsymbol{x}_1^p$) from $\boldsymbol{X}_2$ (or $\boldsymbol{X}_1$), which has the shortest distance to the $\boldsymbol{x}_1$ (or $\boldsymbol{x}_2$), selected in Step 1.

Step 3: Select $\boldsymbol{x}_1^p$ (or $\boldsymbol{x}_2^p$) from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$), which has the shortest distance to $\boldsymbol{x}_2^p$ (or $\boldsymbol{x}_1^p$), selected in Step 2.

When all data are selected from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$) in Step 1, the pairing process is completed. Otherwise, return to Step 1, and repeat the above process. In this process, the same data will not be selected. Finally, the data x_1^p and x_2^p, selected based on the distance, are included in the reduced data set.

If the class boundaries in the data space are based on the distance, the data located close to the boundary can be detected by this method.

3.5 Training and Pairing Method for Training Data Selection

3.5.1 Algorithm

This method combines the training and the pairing as follows:

Step 1: Some number of the training data are randomly selected from $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$. Let the sets of the selected data be $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$.

Step 2: Train the MLNN using the data in $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$.

Step 3: Select the data, with which the network output errors have relatively large error. Let these data be $\boldsymbol{x}_1^e$ and $\boldsymbol{x}_2^e$.

Step 4: Select the data $\boldsymbol{x}_1^p$ and $\boldsymbol{x}_2^p$ from $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$, which have the shortest distance to $\boldsymbol{x}_2^e$ and $\boldsymbol{x}_1^e$, respectively.

Step 5: Select the data $\boldsymbol{x}_1^{pe}$ and $\boldsymbol{x}_2^{pe}$ from $\boldsymbol{X}_1^p$ and $\boldsymbol{X}_2^p$, which have the shortest distance to $\boldsymbol{x}_2^p$ and $\boldsymbol{x}_1^p$, respectively.

A set of $\boldsymbol{x}_1^p$, $\boldsymbol{x}_2^p$ and $\boldsymbol{x}_1^{pe}$, $\boldsymbol{x}_2^{pe}$ will be used in the next training process. Replace the data in $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$ by the new training data, and return to Step 2.

When new data are provided, they are included in $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$. The remaining data of $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ can be also used for this purpose. If the number of the new data is large, some number of the data may be selected, and are included in $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$. After that, return to Step 2.

The data selected in Step 3 satisfy

$$\boldsymbol{X}_1^e = \{\boldsymbol{x}_1 | y(\boldsymbol{x}_1) < a_+, t = 1\} \tag{3.9}$$

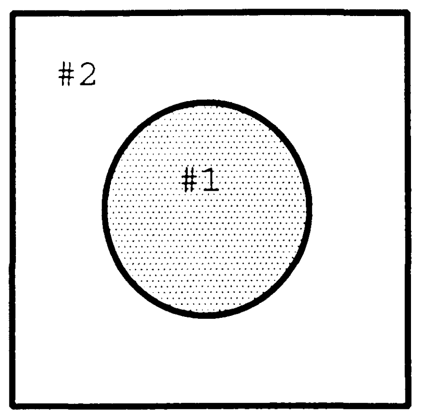
$$\boldsymbol{X}_2^e = \{\boldsymbol{x}_2 | y(\boldsymbol{x}_2) > a_-, t = 0\} \tag{3.10}$$

where $y(\cdot)$ express the output, t is the target, and a_+ and a_- express some levels, for instance 0.7 and 0.3, respectively.

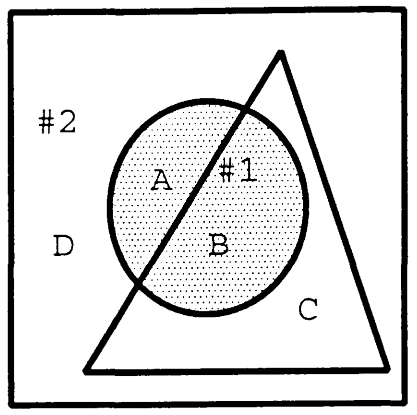
3.5.2 Data Distribution

Purpose of the training in Step 2 is to find the data, which locate close to the class boundary, with less computations. Therefore, the training is stopped at the middle stage in the training process using some criterion. In subsection 3.7.2, this criterion of an off-line training is described. Even though the training is not completely converged, the data, which locate close to the class boundary can be detected using the output error. The details are described in the following.

For convenience, a two-dimensional pattern classification given by Fig.3.1 (a) is employed. It is assumed that the triangle network boundary shown in Fig.3.1(b) is formed in Step 2. The data inside the triangle corresponds to Class 1, and the data outside corresponds to Class 2. In this case, the regions are further divided into A, B, C and D as shown in Fig.3.1 (b). This means that the data locate in B and D are exactly classified into Class 1 and Class 2, respectively. Furthermore, the data in A and C are miss-classified into Class 2 and Class 1, respectively.



(a)



(b)

Figure 3.1: (a) Class distribution, (b) Classification result by not well achieved network.

Following the process in Step 3, the data in A and C will remain due to large output error by miss-classification. Further, the data locate close to the network boundary, in B and D are also detected due to relatively large errors. The error is highly related to the distance from the boundary. However, it is not always proportional to the distance. This will be discussed in Sec.3.7.2. Therefore, the data, with which the output error is relatively large, locate near the network boundary, which is the triangle, at least.

However, the data, which locate close to the network boundary, do not cause large output error. Therefore if the data x_1^e and x_2^e are only selected, the efficient data, which locate close to the boundary, will be missed. Figure 3.2 shows an example, where the data locate in the shaded parts are only satisfy the conditions

Eq.(3.9) and Eq.(3.10), and are detected.

For this reason, the pairing method is combined with the training process. The data in the different classes locate close to x_1^e and x_2^e can be found. They are denoted by x_2^p and x_1^p as shown in Step 4, respectively.

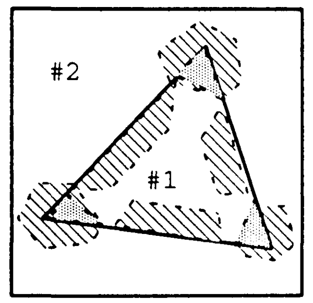


Figure 3.2: Example of two-class classification.

3.6 Training Data Selection in Off-line and On-line Trainings

The proposed data selection method can be applied to both off-line training and on-line training [40]. In the off-line training, all data are given at the beginning of the training. If a large number of training data is available, the data selection is needed to reduce the training time. In the on-line training, the training data are not given all together, but are given successively. Furthermore, they may change continuously. If the data successively received are all accumulated, then the number of the data will be extremely large. Therefore, in this application, the training data selection is important.

3.7 Computer Simulation

3.7.1 Classification Problem and Simulation Conditions

Classification Problem

Two-dimensional two-class classification is employed for computer simulations. The number of input unit N is 2, and the number of output unit K is 2. Then, The data is $\boldsymbol{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2\}$ and the input data is $\boldsymbol{x} = \{x(i), i = 1, 2\}$.

Figure 3.3 shows a concept of the problems. One of the classes is shown as shaded region, and the other is dotted region. White region between the classes shows a gap, so there is no overlap.

In problem 1, two classes are defined as follows:

$$\boldsymbol{X}_1 = \{\boldsymbol{x} \, | x(1)^2 + x(2)^2 \leq (r - \gamma)^2\} \tag{3.11}$$

$$\boldsymbol{X}_2 = \{\boldsymbol{x} \, | x(1)^2 + x(2)^2 > (r + \gamma)^2\} \tag{3.12}$$

here, r is the radius of the circle and is 0.39. γ is the width of the gap, and is 0.02.

In problem 2, two classes are defined as follows:

$$\boldsymbol{X}_1 = \{\boldsymbol{x} \, | A \sin(2\pi \cdot x(1)) \leq x(2) - \gamma\} \tag{3.13}$$

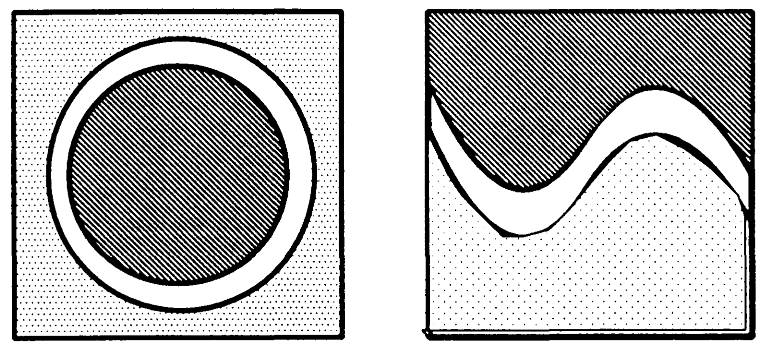


Figure 3.3: Concept of problems. (a) Circle in square, (b) Sinusoidal in square.

$$\boldsymbol{X}_2 = \{\boldsymbol{x} \, | A \sin(2\pi \cdot x(1)) > x(2) + \gamma\} \tag{3.14}$$

where, the A is the amplitude and its value is 0.22.

Simulation Conditions

The number of data for each class is 1000. Six hidden units are used. Two hundreds of data for each class are selected randomly from 1000 data. These are used in following simulations. For the training, learning-rate parameter η is 0.1, and momentum constant α is 0.8. These are decided by experience. Circle in square is called problem 1, and Sinusoidal in square is called problem 2 in the following sections.

3.7.2 Off-line Training

Pairing Method

For off-line training, pairing and training methods are used. Figure 3.4 shows randomly selected data, and Fig.3.5 shows the data found by pairing method. From Fig.3.5, the class boundary is formed by data properly. Sixty-five data are selected for each class.

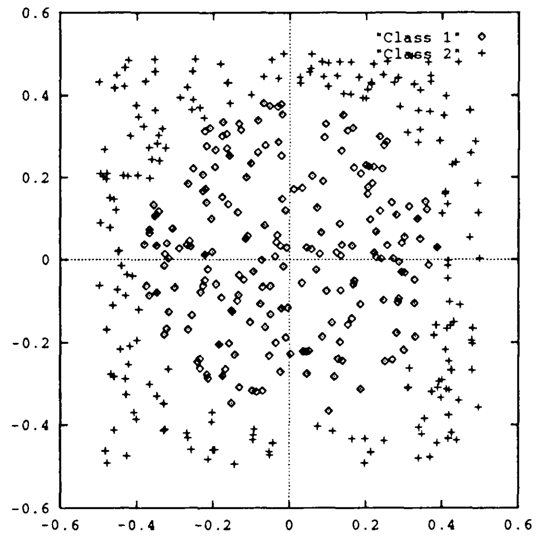


Figure 3.4: Randomly selected data.

The MLNN is trained with selected data. The stopping criterion is 0.001 in the mean square error (MSE) at the output layer. Iteration of 23763 is needed for convergence.

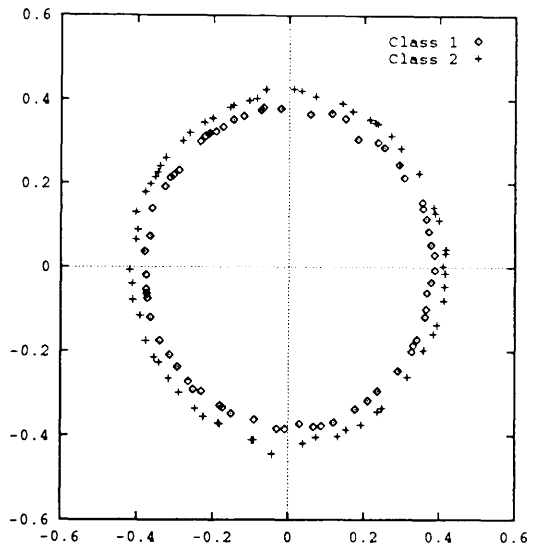


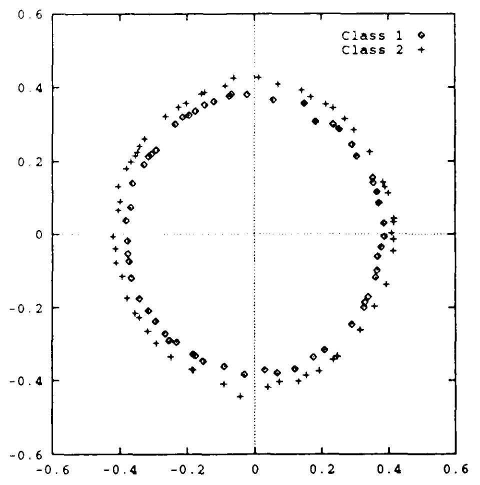
Figure 3.5: Selected data by pairing method.

Training and Pairing Method

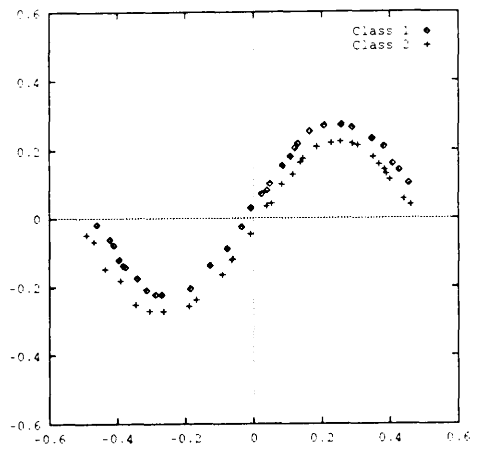
The initial training is stopped at the MSE of ε <0.05. The thresholds, a_+ and a_-, are 1.0, and are equal to ε of 0.073. In problem 1, 207 of data are selected from Class 1, and 164 from Class 2. From Class 1 and Class 2, 116 and 150 of data are selected in problem 2.

Figure 3.6 shows the results. From these figures, the boundary is detected properly.

For stopping the training, four hundreds of validation data are used to have a consistent stopping criterion for conventional method and proposed one. The validation data are subsets of the entire data set. The network output error is calculated



(a)Circle in square.



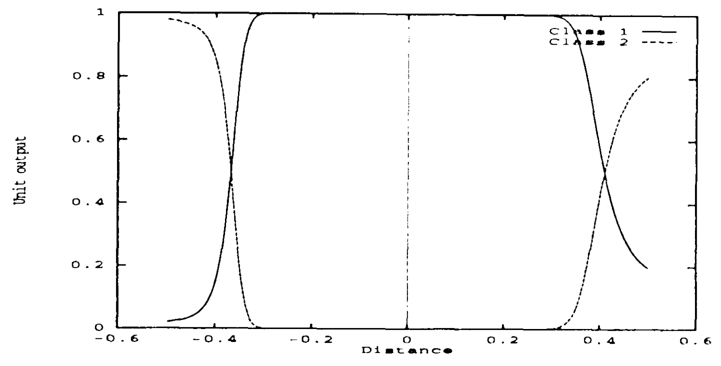
(b)Sinusoidal in square.

Figure 3.6: Selected data

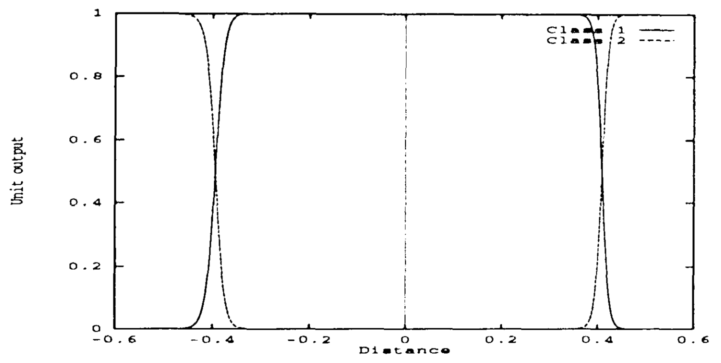
for validation data every iteration. The stopping criterion ε of the validation data is 0.001. Table 3.1 shows the results. In the table, computation of the conventional method is 1.0, and the computation of the proposed method is represented as a ratio of proposed method of conventional one. From this table, the computational complexity is reduced by this process.

Figure 3.7 shows relation among the distance and the output unit output. The MLNN in step 2 of Problem 1 is used. The input data of (a) is $x(2) = 0$ and (b) is $x(1) = x(2)$. In the figures, the horizontal axis is the distance from the origin of the data space to a data. The vertical axis is the output of the output unit to the input data of the horizontal axis. From the figures, (b) has a slope much steeper than (a) near the class boundary, which is at $\pm$ 4.0 in the horizontal axis. Then data

locate on (a) outputs different value from the data on (b) for the same distance as mentioned in subsection 3.5.2



(a) Input data is $x(2) = 0$



(b) Input data is $x(1) = x(2)$

Figure 3.7: Unit output and distance in data space

3.7.3 On-line Training

The on-line training is simulated using partial data of the problems. Problem 1 is used in this simulation. Entire data $\boldsymbol{X}$ is separated into three sets as described below.

$$\boldsymbol{X}_{up} = \{\boldsymbol{x}\,|x(2) \geq 0.167\} \tag{3.15}$$

$$\boldsymbol{X}_{mid} = \{\boldsymbol{x}\,|-0.167 < x(2) < 0.167\} \tag{3.16}$$

$$\boldsymbol{X}_{down} = \{\boldsymbol{x}\,|x(2) \leq -0.167\} \tag{3.17}$$

Each subset data includes 333 data.

$\boldsymbol{X}_{up}$ is used as the training data of Step 1. $\boldsymbol{X}_{mid}$ and $\boldsymbol{X}_{down}$ are used new training data in training of Step 2 of Sec.3.5. The stopping criterion ε in Step 1 is 0.05, and 0.01 for training convergence, respectively. The thresholds for all steps are 1.0.

Figure 3.8 shows the result of on-line training using selected data. The training is converged and their percentage of correctly classified are 100 % for entire data set. The boundary is also detected properly.

3.8 Summary

The training data selection methods used in the MLNN have been proposed. The Pairing method uses the Euclidean distance to find sets of the nearest data to the initially randomly selected data. The training method selects the data based on the network boundary of the MLNN. These methods are combined in this method. Validity of the training methods has been given, and it was confirmed that the training method never lost the data near the class boundary by using pairing method.

Proposed methods have been applied to two applications. One of them is reducing the training of the off-line training, and the other is the on-line training. The training has been converged by using a combination of paring method and training method. The computations to converge the training has been reduced. Training method is also applied to on-line training. In this case, data are selected from the partial data. The training has been converged. Therefore, proposed methods are supported by the simulation results.

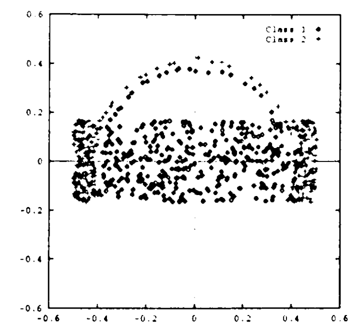
Table 3.1: Comparison of computational complexity between conventional and proposed training.

	Prob. 1		Prob. 2	
	Conv.	Proposed	Conv.	Proposed
Init.	0	134	0	18
Epoch	2444	4394	89	390
N of data	2000	114	2000	62
Comp.	1.0	0.10	1.0	0.14

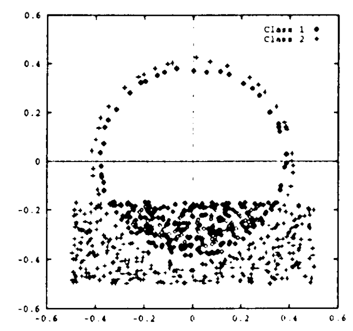
Init.: Epoch of initial training.

N of data: Number of data.

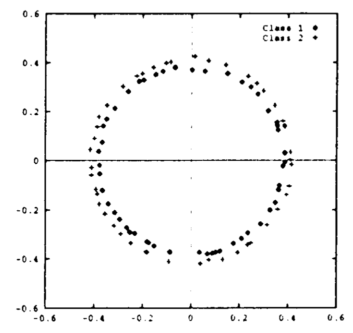
Conv.: Conventional method. Comp. Computation



(a) Training data of Step 1. Upper one third region is selected data and middle region is newly given data.



(b) Training data of step 2. Upper two third regions are selected data. MLNN is trained with data of (a). Rest of the regionvis newly given data.



(c) Selected data of Step 5. MLNN is trained with data of (b).

Figure 3.8: Selected data: Problem 1.

Chapter 4

FREQUENCY SELECTIVE CLASSIFICATION BY MULTILAYER NEURAL NETWORKS

4.1 Introduction

Advantage of multilayer neural networks (NNs) trained by the back-propagation (BP) algorithm is to extract common properties, features or rules, which can be used to classify data included in several groups [2]. Especially, when it is difficult to analyze the common features using conventional methods, the supervised learning, using combinations of the known input and output data, becomes useful.

In this chapter, frequency selective classification by multilayer neural networks (MLNNs) is studied. The signals are classified according to the frequency components included in the signals. Since the frequencies are assigned alternately to several groups, it is difficult to distinguish the waveforms within a short period, and using limited number of samples. The following advantages of the MLNN over conventional methods were confirmed. The neural network can classify the signals using a small number of samples and a short observation period with which the Fourier transform cannot classify. The number of calculations is sufficiently smaller than the convolution calculation, required in digital filters.

A sigmoid function is the one of the most popular activation functions used in the MLNN. However, it is not always optimum. Therefore, properties of activation functions are investigated in this chapter. For this purpose, some typical functions are taken into account. They include a sigmoid function, a radial basis function and a periodic function. They will be compared with each other in classifying multi-frequency signals. Effects of noisy signals will be also discussed in the training and classification processes.

As a result, a rule of thumb for selecting the suitable functions and the combination of several kinds of functions will be provided.

Since the MLNN method is useful for general pattern classification. Therefore, in order to fairly compare the MLNN methods and the linear signal processing methods, the following multi-frequency signals are taken into account. The frequencies are

located alternately between the signal classes, and the amplitude and the phase of each frequency component are generated randomly. Therefore, the signal waveforms of the different classes are similar and similarity between classes is small. This kind of classification may be a difficult problem.

4.2 Multi-frequency Signal

The pth signal class, denoted $\boldsymbol{X}_p$, includes M signals.

$$\boldsymbol{X}_p = \{x_{pm}(n), m = 1 \sim M, n = 0 \sim N-1\} \tag{4.1}$$

The multi-frequency signal is defined as follows:

$$x_{pm}(n) = \sum_{r=1}^{R} A_{mr} \sin(\omega_{pr} nT + \phi_{mr}) \tag{4.2}$$

where, $\omega_{pr} = 2\pi f_{pr}$, f_{pr} is the rth frequency component of the pth class. T is a sampling period. Amplitude A_{mr} and phase ϕ_{mr} of each frequency component are randomly generated in $(0,1]$ and $[0,2\pi)$, respectively. Two classes are used. The number of the signal samples is N=10 or N=20. The frequencies in one class (class 1) are 1, 2 and 3 Hz, and in the other class (class 2), 1.5, 2.5 and 3.5 Hz, respectively. A sampling frequency is 10 Hz. These frequencies can be scaled.

2000 input signals are prepared for each class. For the MLNN, 200 signals are used for training, and 1800 signals for testing. After the training converges, the training signals were perfectly classified. Thus, the MLNN is equivalently evaluated with 2000 signals.

For noisy signals, the additive noise, uniformly distributed in [-0.5,0.5], is used. The SNR is about 6.5 dB.

4.3 Multi-frequency Signal Classification by Using Sigmoid Function

4.3.1 Multilayer Neural Network Design

The MLNN with a single hidden layer is taken into account. Minimizations of the number of hidden units have been well discussed [17, 48]. In this paper, however, it is determined by experience. Almost the highest classification performance was obtained with three hidden units. The number of output units is equal to that of the signal classes. A single output unit is assigned to one class. This means the MLNN is trained so that a single output unit responds to one of the signal classes.

Back-propagation (BP) algorithm is used for training the networks. Both noise-free and noisy signals' sets are used in a training phase and a testing phase. The learning rate η and the momentum term coefficient α are 0.1 and 0.8, respectively, which are decided also by experience. The training is stopped when the mean squared error is less than 0.01 or the number of iterations exceeds 3000.

A ratio of the number of the correctly classified signals and the number of the entire testing signals, defined as "classification rate", is evaluated under several

conditions. A signal is classified into the pth class if the pth output unit takes the maximum value.

4.3.2 Training and Classification

The classification rates are listed in Table 4.1 in percentage. The MLNN can provide high classification rates. The classification rates of using the signals with 20 samples are better than those of the signals with 10 samples. Therefore, non-linearity is notable for 10 samples' signals and is not notable for 20 samples'.

Table 4.1: Probability of exact signal classification in percentage

Methods	N=10		N=20	
	NFS	NS	NFS	NS
MLNN	97.8	86.9	97.7	91.5

N : Number of samples

NFS : Noise Free Signal, NS :Noisy Signal

4.4 Multilayer Neural Network by Using Several Kinds of Activation Functions

4.4.1 Network Structure and Equations

A single-layer neural network is taken into account. N samples of the signal $x_{pm}(n)$ are applied to the input layer in parallel. The nth input unit receives x_{pm}(n). Connection weight from the nth input to the jth hidden unit is denoted w_{nj}. The input and output of the jth hidden unit are given by

$$net_j = \sum_{n=0}^{N-1} w_{nj}x_{pm}(n) + \theta_j \tag{4.3}$$

$$y_j = f_H(net_j) \tag{4.4}$$

Letting the connection weight from the jth hidden unit to the kth output unit be w_{jk}, the input and output of the kth output unit are given by

$$net_k = \sum_{j=0}^{J-1} w_{jk}y_j + \theta_k \tag{4.5}$$

$$y_k = f_O(net_k) \tag{4.6}$$

The activation function of the output layer is the sigmoid function. In the hidden layer, some activation functions include the sigmoid function are used.

The number of output units is equal to that of the signal groups denoted by P. The neural network is trained so that a single output unit responds to one of the signal groups.

4.4.2 Training and Classification

Signals are categorized into training and un-training sets, denoted by X_{T_p} and X_{U_p}, respectively. Their elements are expressed by $x_{T_{pm}}(n)$ and $x_{U_{pm}}(n)$, respectively.

The neural network is trained by using $x_{T_{pm}}$(n), m $= 1 \sim M_T$, for the pth group. Here, M_T is the number of the training data. After the training is completed, the untrained signals $x_{U_{pm}}$(n) are applied to the NN, and the output is calculated. For the input signal $x_{U_{pm}}$(n), if the pth output y_p has the maximum value, then the signal is exactly classified. Otherwise, the network fails in classification.

4.4.3 Selection of Activation Functions

What kinds of activation functions should be selected is very important. At the same time, it is a very difficult problem. In this chapter, the following typical functions are selected for the hidden layer.

When binary target can be considered, then the sigmoid function can be used in the output layer.

Sigmoid function:

$$y_j = f_{sig}(net_j) = \frac{1}{1 + e^{-(net_j)}} \quad (4.7)$$

Sinusoidal function:

$$y_j = f_{sin}(net_j) = \sin(\pi net_j) \quad (4.8)$$

Gaussian function:

$$y_j = f_{gau}(net_j) = e^{-net_j^2} \quad (4.9)$$

The input vectors are distributed in an N-dimensional space. Three functions divide the space as follows:

$$f_{sig}(net_j) \begin{cases} > \alpha_+, & net_j > T_{sig} \\ < \alpha_-, & net_j < T_{sig} \end{cases} \quad (4.10)$$

$$f_{sin}(net_j) \begin{cases} > \alpha_+, & |net_j - (2n\pi + \frac{\pi}{2})| < T_{sin} \\ < \alpha_-, & |net_j - (2n\pi + \frac{3}{2}\pi)| < T_{sin} \end{cases} \quad (4.11)$$

$$f_{gau}(net_j) \begin{cases} > \alpha_+, & |net_j| < T_{gau} \\ < \alpha_-, & |net_j| > T_{gau} \end{cases} \quad (4.12)$$

Here, n is integer.

These space divisions are fundamental, and independent to each other. This is an idea behind selecting the above three functions.

Next step of selecting activation functions is how to combine them. It is also highly dependent on the distribution of the input signals, and is very hard to determine before hand. For this reason, both the homogeneous function and the composite functions are investigated.

4.4.4 Training and Classification

Simulation results are shown in Table 4.2. The training converged using three hidden units for all activation functions. In the case of the Gaussian and the sinusoidal function, the training almost converged with one hidden unit. In this case, noise free signals are used. From this table, the MLNN using the Gaussian activation function achieved the best classification rates. It can classify the un-training signals with only one hidden unit. The MLNN using the sinusoidal activation function achieved worse classification rates than that of the Gaussian activation functions. These two activation functions have similar shape however, differential is non-zero for the sinusoidal function while differential of some part of the Gaussian activation function is zero. This difference will be effect to achieve the classification.

4.4.5 Simulation Using Three Activation Functions

Additive Noise

White noise, denoted *noise(n)*, is generated as random number, and is added to the signal x_{pm}(n). Noisy signal x'_{pm}(n) is given by

Table 4.2: Classification rates by three functions[%]

Activation Function	Hidden Unit	Training		Untraining	
		#1	#2	#1	#2
Sigmoid	1	44.5	100	47.9	100
	3	100	100	97.4	100
Sinusoidal	1	86.0	99.0	79.8	99.0
	3	100	100	92.6	100
Gaussian	1	99.5	100	98.1	100
	3	100	100	99.1	99.9

$$x'_{pm}(n) = x_{pm}(n) + noise(n) \tag{4.13}$$

Training and Classification

The noisy, multi-frequency signals are used for training. N is 10 and M is 200 for each group. After training, un-training signals with white noise are applied, and classification rates are evaluated. White noise is uniformly distributed in the range ± 0.5. The results are shown in Table 4.3. Columns with (A) and (B) list the recognition rates using the training signals without and with white noise, respectively. The MLNN trained without noise is also used for comparison. From these results, it can be confirmed that training using noisy signals is useful to achieve robustness.

Table 4.3: Classification rates using training signals. (A) without and (B) with white noise [%]

Activation Function	Hidden Unit	(A)		(B)	
		#1	#2	#1	#2
Sigmoid	1	47.0	52.9	92.8	28.5
	3	97.3	8.4	82.6	78.0
Sinusoidal	1	80.2	20.9	61.7	87.7
	3	65.9	36.2	79.9	82.7
Gaussian	1	98.2	4.8	71.7	65.9
	3	85.3	46.3	79.8	70.2

Convergence Rates

Figure 4.1 shows learning curves obtained using the three hidden units. The MLNN with the Gaussian function can converge faster than the other. However, the error does not well decreased. The MLNN with the sinusoidal function can also converge faster. At the same time, the error can be well decreased. A convergence rate using the sigmoid function is slow. However, the error can reach to the same level as in using the sinusoidal function.

Learning curve of the sigmoid function is stable after decreasing the network output errors. However, the learning curves of the sinusoidal function and the Gaussian function are unstable. Followings are some analysis of unstable of learning curve of the Gaussian function and the sinusoidal function. Stable and unstable of the learning curve comes from difference of the shape of above functions. The sigmoid function has two saturated regions, so in these regions, small change of the input that produced by modifying the connection weight did not change network output. However, the Gaussian function and the sinusoidal function do not have saturated regions, then small change of the input changes the network output drastically.

4.4.6 Convergence Property Using Single Hidden Unit

Noise Free Multi-frequency Signals

The MLNNs trained without noise are further investigated by hidden unit input and output distribution. Figure 4.2(a) illustrates this distribution, using the sigmoid (a), the sinusoidal (b) and the Gaussian functions (c). One and two follow a, b and c in the figure show the number of hidden units.

In the case of the sigmoid function, the data class 1(#1) and the data class 2(#2) have to be located at the right or the left side. This is a fundamental space division property of the sigmoid function. Thus, the network has to adjust the weights, with which the hidden unit input data are completely separated into the right or the left side. The data #2 is concentrated at the edge of the α_+ as shown in Eq.(4.10), but the data #1 is distributed widely. From this result, the distribution of the hidden unit inputs generated by the multi-frequency signals cannot satisfy the requirements given by Eq.(4.10).

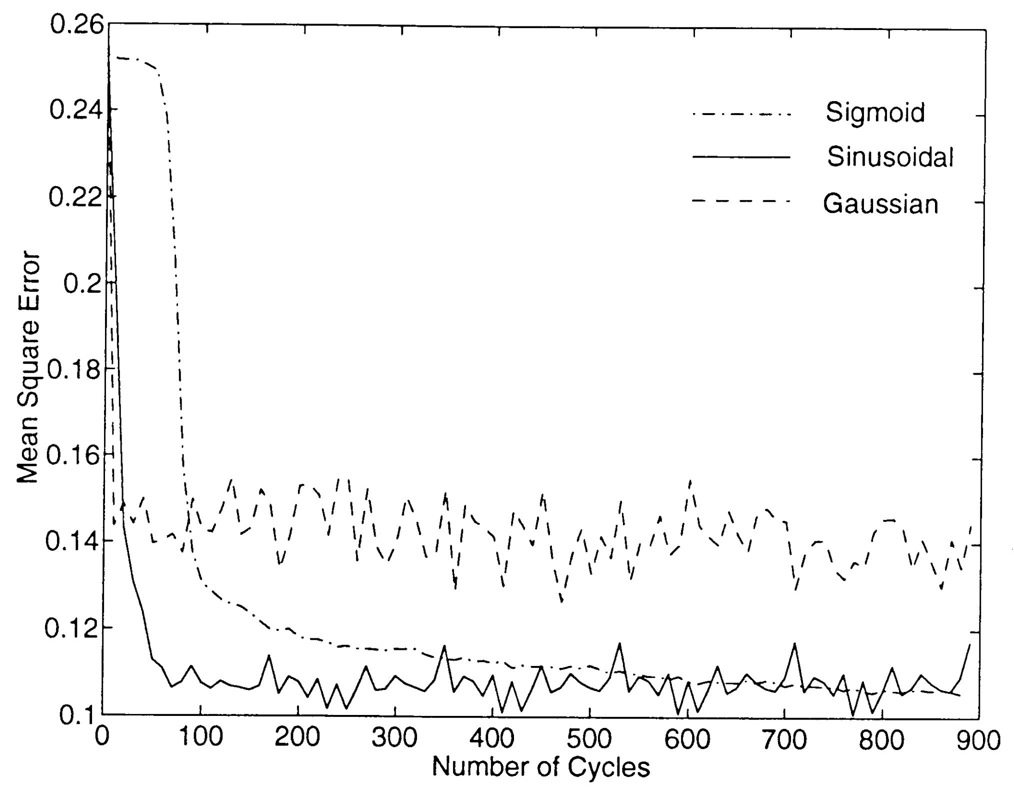
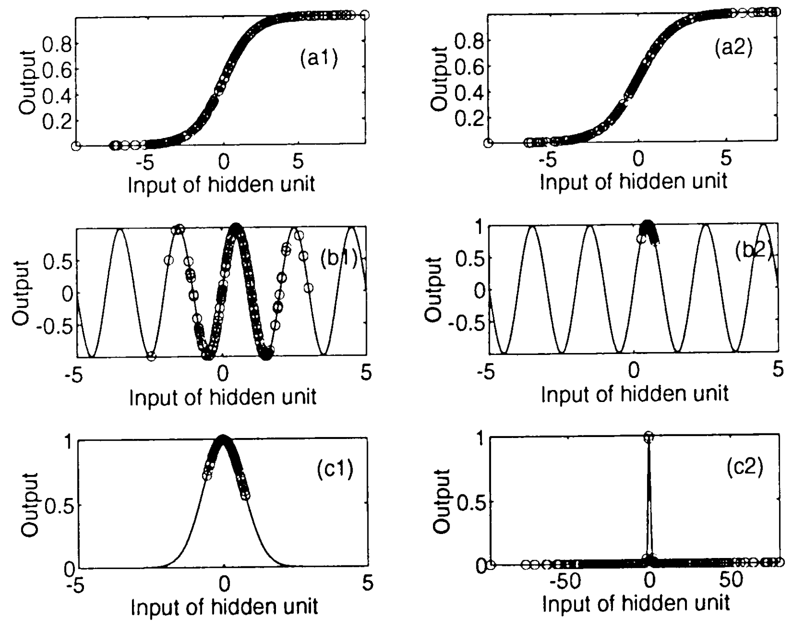
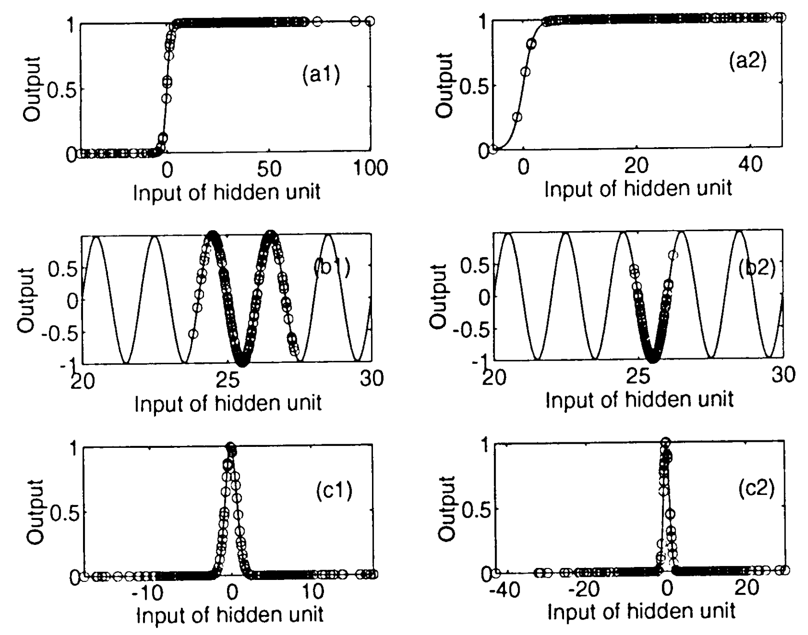


Figure 4.1: Learning curves



(a) Distribution of hidden unit outputs (Trained with noise free signals)



(b) Distribution of hidden unit outputs (Trained with noisy signals)

Figure 4.2: Hidden unit input and output distributions

In the case of the sinusoidal function, the hidden unit inputs of the data #2 locate near one of the peaks and the data #1 distributed widely. The sinusoidal function has large differential coefficient except for the peak. Then the data #2 can be shifted around one of the peaks fast. On the other hand, the data #1 can locate in the region of $f_{sin}(net_j) < \alpha_-$. Therefore, the requirement of the fundamental division property given by Eq.(4.10) is satisfied by the multi-frequency signals.

In the case of the Gaussian function, the data #2 locate around the peak. Differential coefficients around the peak are large, then, the data #2 can be shifted toward this area very fast. Most of the data #1 are distributed both sides.

From these results, the hidden unit inputs of the multi-frequency signals can be concentrated on a narrow range for one group, and the other is distributed widely for the other group.

Thus, the space division property of the Gaussian function is the best match with the distribution of the multi-frequency signals. This function can provide the best accuracy as shown in Table 4.2.

Noisy Multi-frequency Signals

In Figure 4.2 (b), (a), (b) and (c) correspond to the hidden unit inputs and output distributions, in which random noise is added. The network is trained by using the pure multi-frequency signals. After the training, the untrained noisy signals are applied to the MLNN. The distribution of the hidden unit inputs is easily spread by adding the noise.

In the case of the sigmoid, the data #2 distributed widely. However, the most of the data #2 still remains in its own region. Because it has wide stable regions. This is a reason why it can provide better accuracy than the others.

In the case of the Gaussian, the data #2 distributed over the other region, because, a single peak is very narrow. Then these data easily move over the other group's region. Thus, the accuracy is decreased by adding the noise.

The sinusoidal case, the data #2 also widely distributed. However, the sinusoidal function is a periodic function, having several narrow stable regions. Thus, it can provide higher accuracy than that of the Gaussian function.

4.4.7 Convergence Property Using Several Hidden Units

Homogeneous Activation Functions

Figures 4.3, 4.5 and 4.7 show distributions of the hidden unit inputs and outputs. The MLNNs are trained by using the signals without noise. The sigmoid, the sinusoidal and the Gaussian functions are separately used. For each figure, (a), (b) and (c) correspond to one of the hidden unit. (a1), (b1) and (c1) are the response for the data #1, and (a2), (b2) and (c2) are for the data #2.

From these figures, there are two types of distributions, these are concentrated and dispersed distributions. One of two groups is located at near the peak of the functions and the other is widely spread. The overlap of the distributions between the two groups causes miss-classification.

In Fig.4.3, it is very interesting that the data #2 is located at the middle of the slope. Since this region is not a stable region, it can be expected that accuracy is easily degraded by adding the noise. As shown in Table 2, it is true. The classification rates are 97.3% for the data #1 and 8.4% for the data #2. Accuracy for the data #2 is greatly reduced.

Figures 4.4, 4.6 and 4.8 show distribution of the inputs of the two output units. In these figures, (a) and (b) correspond to the data #1 and the data #2, respectively. The region of overlap of the solid and the doted lines will cause miss-classification. We can investigate from these figures, how the hidden units separate the signals into two groups. In the case of #2 data are applied, there is no overlap. So, the hidden unit input space is well separated. In the case of #1 data are applied, there is some overlap. These overlaps cause miss-classification. These results are consistent with the accuracies shown in Table 4.2.

From the figures, the input space of the output units is well separated by the sigmoid and sinusoidal function. So, it can be concluded that three hidden units cooperate to make the distribution of the inputs to the output unit to be linearly separable.

Composite Activation Functions

Three functions can be combined in the same hidden layer. This combination is called 'Composite Activation Function' in this thesis.

Table 4.4 shows classification rates using the multi-frequency signals without noise. In this table, the symbols D through J correspond to the combination of the functions.

The combination C, having three Gaussian functions, achieves the best accuracy. The convergence rate is also the fastest among three functions. The combination D, having all activation functions, achieves better accuracy than the others except for C. However, I and J, which include two Gaussian functions, are worse than D.

K through M are compared with E through J. E and F are better than K. Then adding both the sinusoidal and the Gaussian to the sigmoid can improve the performance. H is better than L, but G is worse than L. Then adding the Gaussian to the sinusoidal can do better than the sigmoid function.

In the most of the combinations, the Gaussian achieves better accuracy. Then, property of each function does not appear straightly in the combinations.

Table 4.5 shows classification rates of the network trained using the noisy signals. Training itself did not converge in all cases. This means that the accuracy is not 100% for all combinations of the functions.

The network using the homogeneous activation function A and B have higher accuracy than the others. However, C does not achieve better accuracy than the others. Then the homogeneous activation function cannot always achieve better accuracy than the composite activation functions.

The network using the composite activation function J has higher accuracy, while C and I have worse accuracy than the others. G and H also provide good accuracy. E and F achieve worse accuracy while A provides good one.

K through M are compared with E through J. G and H are better than L. Then adding the sigmoid or the Gaussian to the sinusoidal works well. K is better than E and F. Then adding both the sinusoidal and the Gaussian to the sigmoid does not work well.

The sinusoidal and sigmoid functions achieve good accuracy in the most of the combinations. However, the sinusoidal combination does not always achieve better accuracy. Thus, property of each function is not straight in the combination, as previously discussed in the no additive noise case.

4.5 Reducing Training Data for Learning Convergence

There are many papers related to data selection method[46, 47, 49], however, in this section, the data selection method to guarantee the generalization performance is investigated through computer simulation. This method is different from the training data selection method that introduced in chapter 3. The data selection method is applied to a multi-frequency signal classification.

4.5.1 Selection of Training Data

In general, by increasing the number of data used for training the MLNN, the generalization performance will be increased. However, difficulty of the network convergence and the computation for the training will be increased. Therefore, in this section, the training data selection method for reducing the training data is proposed and is investigated through computer simulation.

There are two types of the data due to the place in the data space. One is near the class boundary and the other is placed in the class region. From Eq.(2.7), the class boundary is rewritten as $net_k = 0$. Here net_k is kth output unit. Then in all the training data, some data that produce large plus or minus value at the input of the output unit places in the class region. Therefore, at the early stage of the training, these data are important to form the class region, however, after the region is roughly formed, the data near the boundary is important to modify the class boundary. Then in this time, the data that produce large plus or minus values will not be used to modify the class boundary.

The data selection method reduces training data in the training process. Until the mean squared error (MSE) is reached at some threshold E_{th}, all the training data are used for the training. After that, the inputs of the output units are calculated, and the data exceeding the threshold of the absolute value of the input of the output unit Th are removed and the training is done with remained data until MSE of the output units reaches some stopping criterion. Therefore, the network is trained by using the data that are placed near the class boundary. Fig.4.9 shows the training curves. The solid line shows the training using all the training data, and the dotted line and dashed line correspond to the proposed methods. E_{th} is 0.032. From the figure, the training is converged almost the same iteration. However, the number of the data used for training is reduced in proposed methods, then the computation

for the training will be reduced. The threshold of Th=3 is equal to output 0.95 of the output unit. Figure 4.10(a) and (b) show the classification rate of class 1 and 2,(c) shows the number of the data that are reduced. From the figure, when threshold Th=3.0, then 653 of the data are reduced, and the classification rate after the training is almost the same as the one trained without reduceing the number of the data. From above results, it can be concluded that the reduced data in the training process are not useful to modify the class boundary. Therefore, usefulness of this method is proved.

4.6 Summary

Properties of the activation functions for multi-frequency signal classification have been discussed using multilayer neural network supervised by BP algorithm. The Gaussian function can provide the highest performance for the signals without noise. However, it is sensitive to the additive noise. The sigmoid function is not useful for a single hidden unit. If several hidden units are used, then the sigmoid function becomes useful, and is insensitive to the additive noise. The sinusoidal function is useful for noisy signal.

Moreover, the training data selection method is proposed. By using this method, the classification rates are the same as the one trained by using all the data. The number of training computations is reduced. Therefore, the proposed method guarantees the generalization performance and at the same time, reducing the number of training computations.

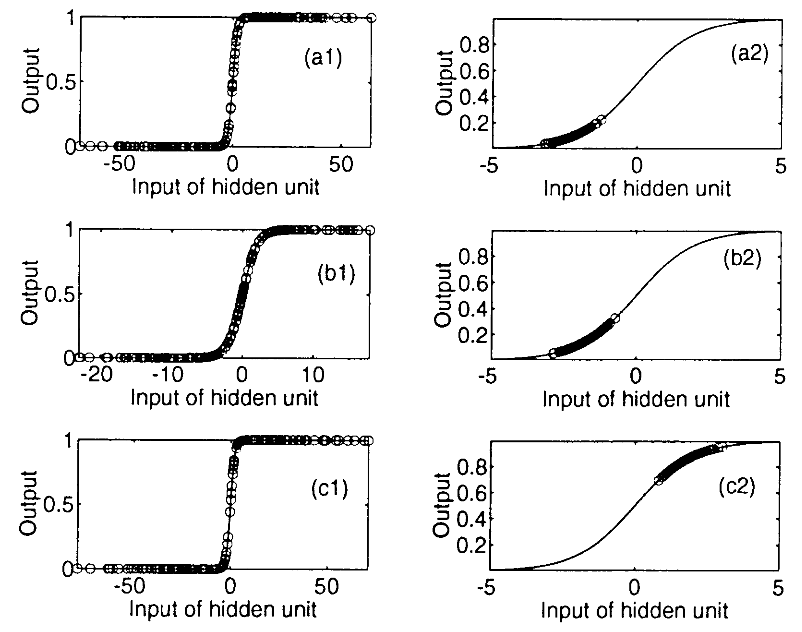


Figure 4.3: Distribution of sigmoid hidden unit outputs

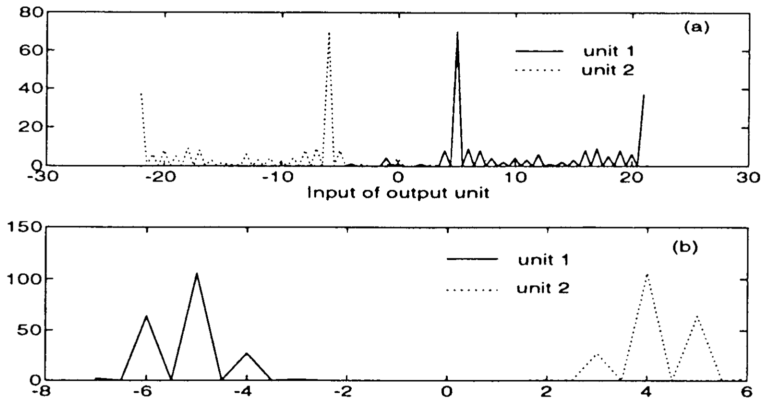


Figure 4.4: Distribution of output unit inputs

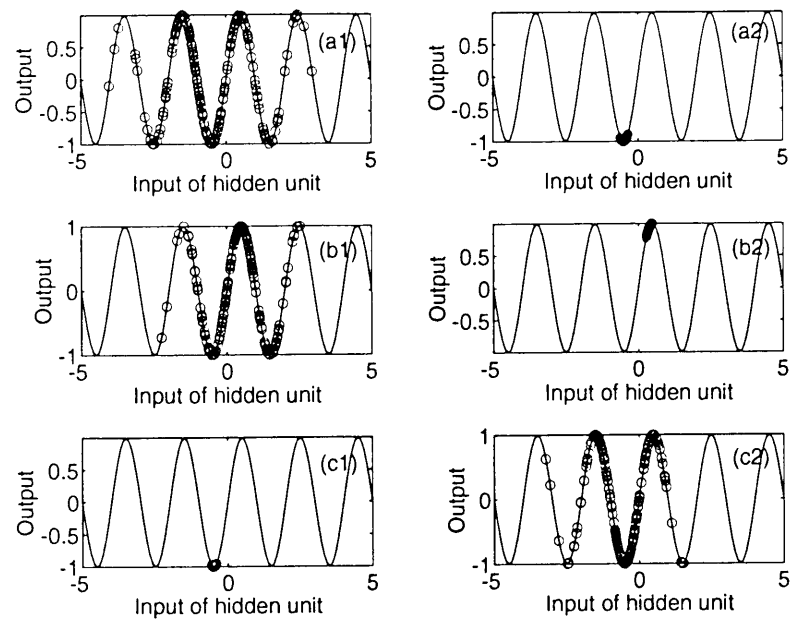


Figure 4.5: Distribution of sinusoidal hidden unit outputs

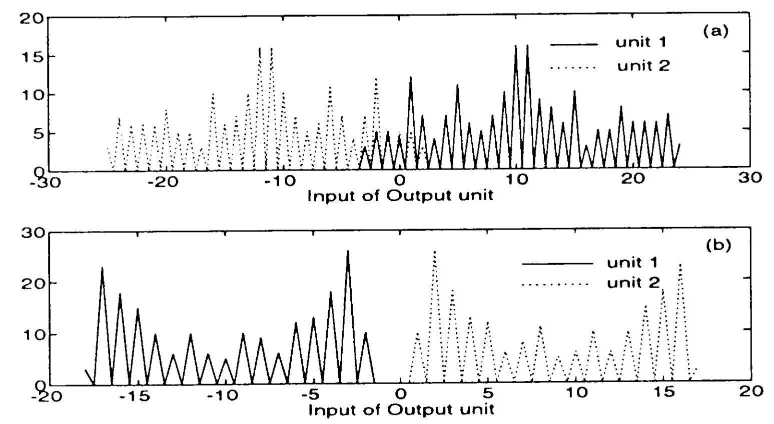


Figure 4.6: Distribution of output unit inputs

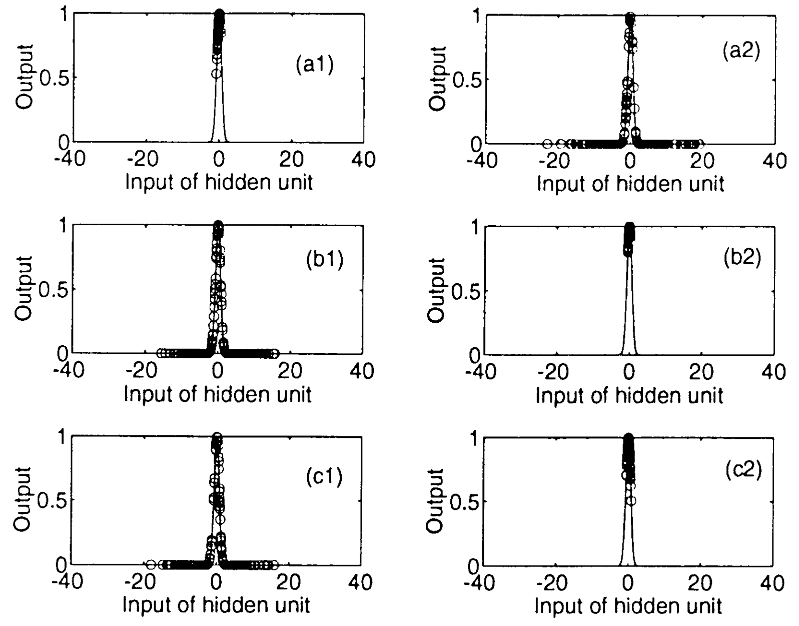


Figure 4.7: Distribution of Gaussian hidden unit outputs

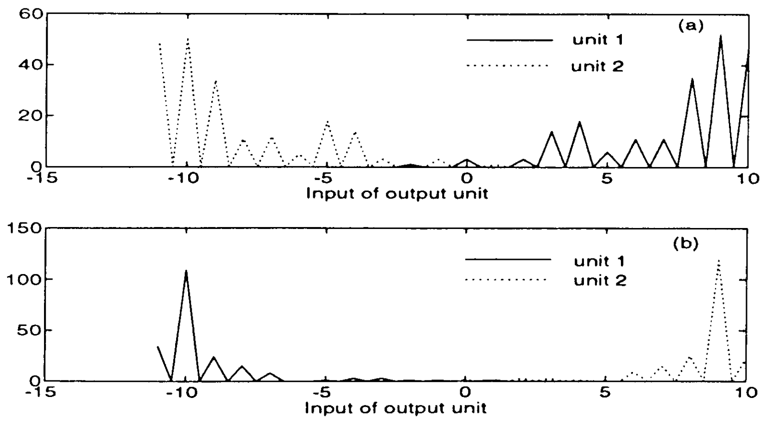


Figure 4.8: Distribution of output unit inputs

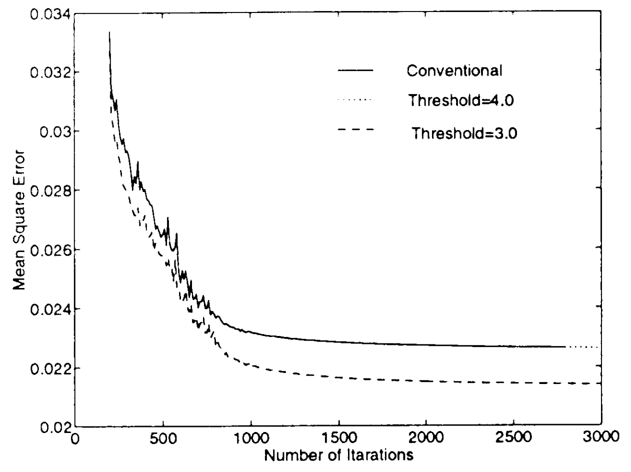


Figure 4.9: Learninig Curve

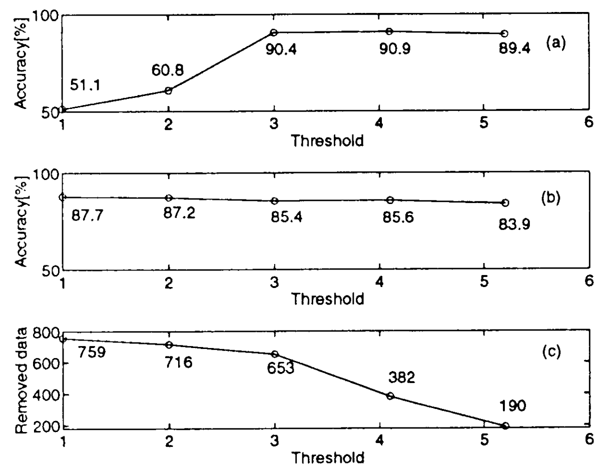


Figure 4.10: Th and classification rate and the number of data which are reduced

Table 4.4: Classification rates using signals without noise

	Combination			Training		Untraining		
	Sig	Sin	Gauss	#1	#2	#1	#2	Ave.
A	3	0	0	100	100	97.4	100	98.7
B	0	3	0	100	100	92.6	100	96.3
C	0	0	3	100	100	99.1	99.9	99.5
D	1	1	1	100	100	100	98.3	99.1
E	2	1	0	99.5	100	96.6	98.4	97.5
F	2	0	1	100	100	97.4	100	98.7
G	1	2	0	93.5	98.5	83.8	97.3	90.6
H	0	2	1	100	100	99.9	97.8	98.9
I	1	0	2	100	100	96.2	99.6	97.9
J	0	1	2	100	100	97.3	98.9	98.1
K	2	0	0	99.0	100	94.0	100	97.2
L	0	2	0	86.0	95.5	86.8	97.3	92.1
M	0	0	2	99.5	98.5	99.4	98.8	99.1

Table 4.5: Classification rates using signals with noise

	Combination			Training		Untraining		
	Sig	Sin	Gauss	#1	#2	#1	#2	Ave.
A	3	0	0	83.5	86.0	82.6	78.9	80.8
B	0	3	0	84.5	89.0	79.9	82.7	81.3
C	0	0	3	87.0	81.5	79.8	70.2	75.0
D	1	1	1	77.0	92.5	69.1	84.3	77.6
E	2	1	0	88.5	77.0	80.9	67.8	74.4
F	2	0	1	78.5	98.5	63.8	85.9	74.9
G	1	2	0	74.0	92.5	69.4	87.0	78.2
H	0	2	1	79.0	92.5	72.3	84.3	78.3
I	1	0	2	84.0	87.5	73.5	75.9	74.7
J	0	1	2	84.5	82.0	81.0	78.5	79.8
K	2	0	0	91.5	70.5	81.3	69.3	75.3
L	0	2	0	80.3	83.0	79.1	73.6	76.4
M	0	0	2	75.5	85.0	74.6	76.1	75.4

Chapter 5

PATTERN CLASSIFICATION BY LINEAR SIGNAL PROCESSING METHODS

5.1 Introduction

The classification by the linear signal processing (LSP) methods are introduced in the pattern classification point of view. The classification mechanism and its classification performance are discussed. The LSP methods are not a pattern classification method, however, they are used in the process that can be seen as a pattern classification. In the analysis in this chapter, the signal consists of N samples is seen as the N-dimensional vector. As the same as the pattern classification by the

MLNN, the pattern classification is regarded in the same light as dividing the N-dimensional space into the class regions to suit the distribution of the signals. Then, the classification is performed based on the distance between the filter coefficients and the patterns.

5.2 Linear Signal Processing Methods

5.2.1 Pattern Matching Methods

A classification by using pattern matching method is carried out to measure a distance from a template pattern to an input pattern, and the input pattern is classified into the class that the nearest template is included. This method is called the nearest neighbor method[50]. The Euclidean distance and the Maharanobis generalized distance(MGD)[51] can be used as the measurement of a distance. In this thesis, the nearest neighbor method using the Euclidean distance is called by Euclidean method, and the nearest neighbor method using the MGD is called by the MGD method. In both of using the Euclidean distance and the MGD, each class' template forms a subclass. In the case of small number of templates are used, templates roughly cover the class region and form the class boundary. Then, mis-classification will be occurred.

The classification performance by the Euclidean method will be dropped when the distribution of the templates is biased. On the other hand, the MGD measures the distance from the central vector of the pth class templates μ_p to the input signal

$\boldsymbol{x}$ by

$$\hat{d}_p^2 = (\boldsymbol{x} - \boldsymbol{\mu}_p)' \boldsymbol{C}_p^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_p) \tag{5.1}$$

Here, $\boldsymbol{C}_p$ is the covariance matrix of p class signals. The central vector is a mean vector of the signals that correspond to the training signals of the MLNN. $\boldsymbol{x}'$ is the transpose of a vector $\boldsymbol{x}$. From the equation above, the MGD normalizes the distance $(\boldsymbol{x} - \boldsymbol{\mu}_p)^2$ by the covariance matrix of the templates, so the classification performance is robust against the placement of the templates.

The k-mean clustering [52, 53] and the Gaussian Classifier [50] are famous pattern classification methods. The k-mean clustering performs as the same as the Euclidean method using many templates. The Gaussian classifier can be considered as a single layer linear perceptron. It measures the distance by the MGD, and estimates the joint-probability density of the input data as the Gaussian distribution. Then, this method has the same classification performance as the MGD method using many templates. Due to the reasons above, the Euclidean method and the MGD method are employed as the pattern classification methods.

5.2.2 Frequency Analysis Methods

The Fourier transform and the filters[54] are useful as the signal classification method using frequency analysis. By using the Fourier transform, if the frequency component of the pth class is the maximum among classes, then the signal is classified into the pth class. The classification by the filters, a filter bank that consists of the filters for each class, is used to identify the input signal class. If the pth class filter output in power is the maximum among the filters, the input signal is classified into the pth class. In both cases, the Fourier kernel or the filter coefficient is designed to extract specified frequency component. Therefore, pre-processing is needed to estimate the key frequency information to achieve the classification.

The Fourier kernel and the filter coefficient are correspond to the template pattern of the pattern classification. However, the Fourier kernel and the filter coefficients are designed, so degree of freedom to select these coefficients are lower than selection of template of the pattern classification. More detailed discussion is given in Sec. 5.3.

Fourier Transform

For the Fourier transform, the frequency component of the input signal $\{x(n), n = 0 \sim N-1\}$ or $\boldsymbol{x}$ is extracted by calculating the inner product of the input signal $\boldsymbol{x}$ and the Fourier kernel $\{e^{-j\omega n \mathrm{T}}, n = 1 \sim N\}$.

$$X(\omega) = \sum_{n=0}^{N-1} x(n) e^{-j\omega nT}. \tag{5.2}$$

Here, T is the sampling period and $\omega = 2\pi f$. Then, the Fourier kernel is corresponding to the template pattern of the pattern classification method.

Finite Impulse Response Filter

Finite Impulse Response (FIR) filter with a direct form[54], the output signal of the FIR filter is calculated by Eq. (5.3) in the steady state.

$$y_p(n) = \sum_{k=0}^{N-1} x(k+n_0) h_p(n-k), \qquad h_p(n-k) = 0, n-k < 0 \tag{5.3}$$

Here, $h_p(n-k)$ are the filter coefficients, by which the pth class signal can be extracted. This type of the FIR filter is called as FIR1 in this thesis.

The signals can be also detected by suppressing the class frequencies. This type of the FIR filter is called FIR2 in this thesis. The transfer function of the pth class, denoted $H_{sup_p}(\mathrm{z})$, has zeros on the unit circle at the corresponding frequencies.

$$H_{sup_p}(\mathrm{z}) = h_0 \prod_{k=1}^{K} (1 - 2\cos\omega_{pk} \mathrm{T} \mathrm{z}^{-1} + \mathrm{z}^{-2}) \tag{5.4}$$

where h_0 is a constant and ω_{pk} is the frequency components included in pth class. The output is calculated by using Eq.(5.3). The order of the transfer function of the FIR2 is as the same as the number of the samples of the signal.

Infinite Impulse Response Filter

An infinite impulse response (IIR) filter [54] requires a low-order transfer function, which are a small number of coefficients. However, the recurrent structure requires higher computation than the FIR filter.

One of the IIR filter realization is a cascade form of the second-order circuits, whose transfer function is written as

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}. \tag{5.5}$$

The output $y(n)$ is calculated as

$$w(n) = x(n) - b_1 w(n-1) - b_2 w(n-2) \tag{5.6}$$

$$y(n) = a_0 w(n) + a_1 w(n-1) + a_2 w(n-2). \tag{5.7}$$

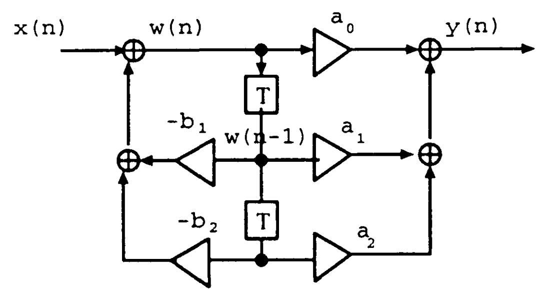


Figure 5.1: Second order IIR filter.

$w(n)$ is an internal variable as shown in Fig.5.1. A high-Q filter can be realized using a low-order transfer function. However, the linear phase response cannot be guaranteed.

5.2.3 Spectrum Estimation Methods

Maximum Entropy Method(MEM)[54], which is a spectrum estimation method, estimates an auto-regressive (AR) model of the signals. The main benefit of using

this method is that it can estimate an interest spectrum using the limited number of signal samples.

From the Wienner-Khinchin's law, the spectrum is equal to the Fourier transform of the auto-correlation of the signal. The power spectrum $P(\omega)$ is given by

$$P(\omega) = \sum_{i=-M}^{M} \gamma_i z^i. \tag{5.8}$$

Where, γ_i is the auto-correlation sequence of the signal x(n) with i lag. M is the order of the filter. The power spectrum $P(\omega)$ is modeled by

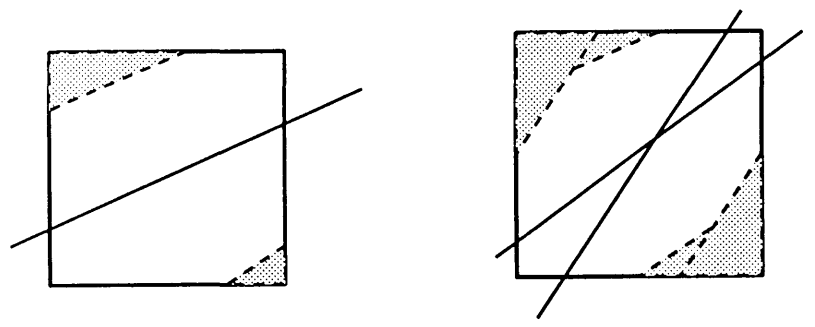
$$P(\omega) \approx \frac{a_0}{\left|1 + \sum_{k=1}^{M} a_k z^k\right|^2}, \tag{5.9}$$

a_0 and $\{a_k\}$ are unknown coefficients of the prediction-error filter. These coefficients are obtained by Eqs.(5.8) and (5.9). Akaike showed limit of M as $M < (2 \sim 3)\sqrt{(N)}$ [55]. If M is less than above, false peaks will not be appeared. The classification is as the same as the FIR filters.

Another method in this category is a super-resolution algorithm (Multiple Signal Classification: MUSIC)[56]. The MUSIC is used to estimate frequencies and directions of waves arrive at the uniformly spaced linear sensor array. The number of sensors is limited. Usually, the MUSIC allows around 30dB of SNR(Signal Noise Ratio) [4]. This SNR is smaller than that used in the simulation in chapter 4 and 6, since, maximum entropy method is employed.

From above discussions, the following methods are employed as the LSP methods: The pattern matching method using the Euclidean distance and the Maharanobis generalized method; the frequency analysis method using the Fourier transform and the filters; and the spectrum estimation method using the maximum entropy method.

5.3 Analysis of Degree of Freedom to form Detection Regions



(a) FIR filter coefficients and output samples distributions. (Small number of samples are used)

(b) FIR filter coefficients and output sample distributions. (Many samples are used)

(a)N is large and k_l is small. (b) N is small and k_h is large.

Figure 5.2: Signal detection region of FIR filter.

Classification performance of the LF methods is investigated based on the spectrum distribution of the signals regarding the number of the signal samples.

When many samples are used to represent the input signals, the frequency components are almost the same as the original signal's. Then highly accurate signal classification is possible by the LF methods. To analyze a frequency component by a high-Q bandpass filter (BPF), difference of the output power between the input

signals that include or not include the frequency components are obtained. Moreover, the output of the high-Q BPF nearly regarded as a sinusoidal waveform. Then it is possible to identify that the frequency component is included in an input signal or not with small number of filter output samples.

If the input signal is in the pth class, the outputs of the pth class filter $y_p(n)$ and the others $y_{p'}(n)$ satisfy the following equation.

$$\sum_{n=n_1}^{n_1+K_l-1} |y_p(n)| \gg max\sum_{n=n_1}^{n_1+K_l-1} |y_{p'}(n)| \qquad (5.10)$$

Where, K_l is the number of the filter outputs, and is assumed to be small. Supposing an appropriate threshold α, this condition can be replaced by

$$\sum_{n=n_1}^{n_1+K_l-1} |y_p(n)| = \sum_{n=n_1}^{n_1+K_l-1} |\sum_{k=0}^{N-1} x(k+n_0)h_p(n-k)| > \alpha. \qquad (5.11)$$

In this equation, the right hand inequality forms some regions in an N-dimensional space, where the pth class signals are included. This region is called a signal detection region of the pth class. Figure 2 (a) shows a conceptual image of the signal detection regions given by Eq.(5.11) for two-dimensional signals. The shaded parts are the signal detection regions and the solid line shows a boundary of the regions that formed by $y_p(n) = 0$ in Eq.(5.11). The signals of the pth class are concentrated in the shaded parts and the other class signals are distributed around the boundary.

When the number of the signal samples is small, the frequency components or the spectrum distribution is distorted from those of the original signal's. Because, using a small number of the signal samples is equal to using a short interval window, and this affects the amplitude response of the signal. The signal detection region is formed by

$$\sum_{n=n_1}^{n_1+K_h-1} |y_p(n)| > \alpha. \tag{5.12}$$

Where $K_h > K_l$. The region specified by this inequality is wider than that given by Eq.(5.11). Equation (5.12) can be satisfied when some outputs take large values than α. Then the condition of the classification is relaxed by using many output samples. A conceptual image of this extended regions is illustrated as some shaded parts in Fig.2(b).

5.3.1 Signal Classification by Output Power

The same number of the filters as that of the signal classes is used in the signal classification. The pth class filter is designed to extract the frequency components of this class, and to suppress those of all the other classes. The power of the pth filter output S_p is calculated by

$$S_p = \sum_{n=n_1}^{n_1+K-1} y_p^2(n). \tag{5.13}$$

Where, $y_p(n)$ is the filter output and K is the number of the output samples. n_1 is the beginning of the steady state response. Classification is done by using the following criterion.

$$\text{If} \quad S_p = \max_{p'}\{S_{p'}\} \quad \text{then} \quad \boldsymbol{x} \in \boldsymbol{X}_p \tag{5.14}$$

that is the signal is classified into the pth class.

Next, computation complexity required in calculating the output power is discussed. The FIR filter with a direct form always needs N computations in calculating one output as shown in Eq.(5.3). One computation includes one multiplication and one addition. It is independent of the filter order denoted N_{FIR}. In other words, a very high-Q, which is high-order FIR filter can be used to achieve higher resolution without increasing in the memory capacity and the number of computations. The output samples in the steady state are used in calculating the output power.

On the other hand, the IIR filter has a recursive structure as shown in Fig.5.1. In calculating the Mth output $y(M)$, the filter should operates from $n = 0$ to $n = M$. Letting filter order be N_{IIR}, $y(M)$ requires $(5/2)N_{IIR}M$ computations. It is mainly determined by N_{IIR} and M, not N. Here, we assume the 2nd-order section needs five computations as shown in Fig.5.1. Furthermore, $y(M)$ in the steady state should be used in estimating the output power. Thus, even though $N_{IIR} \ll N_{FIR}$, the IIR filter may require more computations than the FIR filter in estimating the output power.

5.3.2 Degree of Freedom of Space Division

The degree of freedom of forming the class region is discussed in the following. The filter coefficients used to calculate $y(n)$ are $h_p(n - N + 1) \sim h_p(n)$. Thus, a set of successive N coefficients is used to calculate $y(n)$. Let this set be $\boldsymbol{h}_p(n, N)$. There is strong correlation among $\boldsymbol{h}_p(n, N)$. In other words, they cannot be determined independently. They are designed to extract the necessary frequencies. $\boldsymbol{h}_p(n, N)$ corresponds to a set of the connection weights from the input to the hidden layers. These connection weights do not have any constraints. They can be adjusted using the training data. Therefore, the MLNN can realize more flexible subregions, and is superior to the LF method in pattern classification. However, the MLNN is dependent on the training data. The training should be done to achieve good generalization performance.

Discussions based on computer simulation will be given in Sec.6.5 and Sec.6.6.

5.3.3 Correlation of Partial Coefficients of Filter

The input of the FIR filter is denoted by $x(n)$, its coefficients are denoted $h(n)$. Then, the output of the filter $y(n)$ is given by the next equation.

$$y(n) = \sum_{k=0}^{N-1} x(k)h(n-k) \tag{5.15}$$

Here, $x(n)$ consists with N samples of $n = 0 \sim N-1$. To get $y(n_0)$, the coefficients of $h(n_0) \sim h(n_0 - N + 1)$ are used, and for $y(n_0 + m)$, $h(n_0 + m) \sim h(n_0 + m - N + 1)$ are used. Denote above as $h_0(n)$ and $h_m(n)$, and by using time domain window $w(n)$, they can be rewritten as follows.

$$h_0(n) = h(n)w(n_0 - n), n = 0 \sim N-1 \tag{5.16}$$

$$h_m(n) = h(n)w(n_0 + m - n), n = 0 \sim N-1 \tag{5.17}$$

$$w(n) = \begin{cases} 1, & 0 \leqq n \leqq N-1 \\ 0, & n < 0, N \leqq n \end{cases} \tag{5.18}$$

First, narrow bandwidth lowpass filter (LPF) is discussed[57]. The passing-band denotes f_c, the sampling frequency is f_s, the amplitude response at passing-band is unity, and stopping-band is zero. Then the impulse response of the LSP $h(n)$ is given by the next equation.

$$h(n) = \frac{1}{2\pi}\int_{-\omega_c T}^{\omega_c T} e^{j\omega n T} d\omega T, \; \omega_c = 2\pi f_c \tag{5.19}$$

$$= \frac{2f_c}{f_s}\frac{\sin \omega_c n T}{\omega_c n T}, \; \omega_c n T = 2\pi(\frac{f_c}{f_s})n \tag{5.20}$$

When the order of the filter is $N_f - 1$, then the coefficient $h(n)$ shifted $(N_f - 1)/2$ to the right is used for the interval of $0 \sim N_f - 1$.

Next, the similarity of $h_0(n)$ and $h_m(n)$ is discussed. From Eq.(5.20), to compare $\omega_c n_0 T$ and $\omega_c(n_0 + m)T$, the difference of these two is $\omega_c m T = 2\pi(f_c/f_s)m$. In general, for narrow bandwidth LPF, $f_s/f_c \gg 1$ then, for some small m that satisfy $f_s/f_c \gg m$ $2\pi(f_c/f_s)m \ll 2\pi$ is true and $h_0(n)$ and $h_m(n)$ become similar.

Moreover, the filter coefficients of the FIR filter or the impulse response narrow bandpass filter (BPF) (denotes $h_p(n)$) is discussed. In approximately, $h_p(n)$ is given by the impulse response of the narrow Bandwidth LPF multiply the sinusoidal waveform whose center frequency is at the passband. Here, several band is considered, and the center frequencies are f_1, f_2, f_3, respectively. Then $h_p(n)$ is written as follow.

$$h_p(n) = h(n)\{\cos(2\pi f_1 nT) + \cos(2\pi f_2 nT) + \cos(2\pi f_3 nT)\} \tag{5.21}$$

As the same as the former discussion, the filter outputs of $y(n_0)$ and $y(n_0 + m)$ are calculated by using the partial coefficients of the filter.

$$h_{p0}(n) = h_p(n)w(n_0 - n) \tag{5.22}$$

$$h_{pm}(n) = h_p(n)w(n_0 + m - n) \tag{5.23}$$

In this case, f_1, f_2, $f_3 \ll f_s$ is not always true, then the similarity of the filter coefficients is not guaranteed. Therefore, as the waveform itself, $h_{p0}(n)$ and $h_{pm}(n)$ are not similar. However, the sinusoidal waveform to generate the $h_p(n)$ has a high correlation among the partial waveforms, so the correlation of $h_{p0}(n)$ and $h_{pm}(n)$ will be high. In other word, in the range of $f_s/f_c \gg m$ or $h_0(n) \risingdotseq h_m(n)$ is true, the correlation of $h_{p0}(n)$ and $h_{pm}(n)$ is the same as the correlation between the samples of the sinusoidal wave.

When m is relatively large and the correlation of wave form of $h_0(n)$ and $h_m(n)$

become lower, the correlation of $h_{p0}(n)$ and $h_{pm}(n)$ becomes lower, however, they have some correlation.

As discussed above, the correlation of the partial coefficients of the narrow BPF filter is high, therefore, the degree of freedom to form a class region in the N-dimensional space by partial coefficients of the filter is low.

5.4 Summary

In this chapter, the classification performances of the LSP methods are analyzed based on their capability of forming the class regions related to the signal distribution in the N-dimensional space.

From the analytical results, the pattern classification performance is related to the number of the samples of the signal. When the signal consists of many samples, the orthogonality of the frequency components of the signal is guaranteed, then the output power of the filter for the signal that does not include the extracting frequency components is always relatively small. Then the classification is possible by using the small number of the output sample of the filter. On the other hand, the number of the samples is small and the orthogonality of the frequency components is not guaranteed, the output power of the filter for the signal does not include the extracting frequency components is not always small and many output samples of the filter need to do exact comparison.

Chapter 6

COMPARISON BETWEEN MULTILAYER NEURAL NETWORKS AND LINEAR SIGNAL PROCESSING METHODS

6.1 Introduction

In chapter 2, the pattern classification ability of the MLNN is analytically Investigated and it has been pointed out that the MLNN has a large degree of freedom to form the class region in an N-dimensional space.

In chapter 5, the pattern classification performances of the LSP methods are investigated theoretically. From the analytical results, the filter methods design its coefficients to extract specific frequency components. So, the degree of freedom of selecting filter coefficients is small.

These analytical results claimed that the MLNN has a superiority to form the class region in an N-dimensional space, however, the classification performances of the MLNNs and the LSP methods are not investigated. To make clear the superiority of these two, in this chapter, the signal selective classification performances of the MLNNs and the LSP methods are compared through computer simulations. The comparison is curried out from several points of view; classification rates, number of the signal samples and the computational complexity of the methods. The classification problems used here are the multi-frequency signal classification (refer to section 4.2) and the dial-tone recognition.

6.2 Multilayer Neural Networks

The network structure is as the same as used in the chapter 4. Minimization of the number of the hidden units has been well discussed [17, 48]. In this chapter, however, it is determined by experience. Almost the highest classification performance was obtained with three hidden units. The number of output units is equal to that of the signal classes. A single output unit is assigned to one class. This means the

MLNN is trained so that a single output unit responds to one of the signal classes.

Back-propagation (BP) algorithm is used for training the networks. Both noise-free and noisy signals' sets are used in a training phase and a testing phase. Noise used in this chapter is as the same as the one used in Sec. 4.2. The learning rate η and the momentum term coefficient α are 0.1 and 0.8, respectively, which are decided also by experience. The training is stopped when the mean squared error is less than 0.01 or the number of iterations exceeds 3000.

A ratio of the number of the correctly classified signals and the number of the entire testing signals, denoted "classification rate", is evaluated under several conditions. A signal is classified into the pth class if the pth output unit takes the maximum value.

6.3 Design and Classification of Linear Signal Processing Methods

6.3.1 Design of Linear Signal Processing Methods

When the frequency components of the signals are known in advance, the filter specification can be determined, and the filters can be designed to extract the necessary frequencies and suppress the unnecessary ones. Usually, high-Q amplitude and linear phase are desirable. On the other hand, when the frequencies are not known, the filters cannot be designed following some specifications, rather they should be designed through some training algorithms like "adaptive filters". In this paper, however, it is assumed that the frequency components of the signals are known, and the former case is taken into account.

(1) FIR Filter 1 (FIR1)

Figure 6.1 shows an example of the amplitude response of a 1000th-order FIR filter for the class 1. It has the peaks at frequencies 1, 2 and 3 Hz, and the bandwidth is 0.02 Hz. For class 2, the amplitude response that has the peaks at class 2 frequencies is used.

The FIR filter with a direct form [54] is used, the output signal of the FIR filter is calculated by Eq. (5.3) in the steady state. As discussed in section 5.3.1, a very high-Q, which is high-order FIR filter can be used to achieve higher resolution without increasing in the memory capacity and the number of computations. The output samples in the steady state are used in calculating the output power. A linear phase is easily realized.

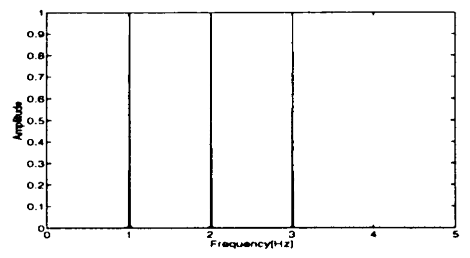


Figure 6.1: Amplitude response of FIR1 filter designed to extract class 1 signals.

(2) FIR Filter 2 (FIR2)

The order of the transfer function of the FIR2 is as the same as the number of the samples of the signals. The order is nine and nineteen for $N = 10$ and $N = 20$, respectively. The zero frequencies are located on the unit circle.

(3) IIR filter

The transfer function of the ith second order circuit is given as follows:

$$H_i(z) = \frac{1 - 2\cos\theta_{zi} z^{-1} + z^{-2}}{1 - 2r_i \cos\theta_{pi} z^{-1} + r_i^2 z^{-2}}. \qquad (6.1)$$

Here, r_i is the magnitude of the ith poles. r_i is less than one. θ_p and θ_z are the pole and the zero frequencies, respectively. They are given as next equations.

$$\theta_{pi} = 2\pi\frac{f_{pi}}{f_s}, \qquad i = 1, 2, \ldots \qquad (6.2)$$

$$\theta_{zi} = 2\pi\frac{f_{zi}}{f_s}, \qquad i = 1, 2, \ldots \qquad (6.3)$$

$$(6.4)$$

where, f_s is the sampling frequency.

The total transfer function is

$$H(z) = \prod_{i=1}^{I} H_i(z). \qquad (6.5)$$

In order to realize a high-Q filter, fifteen zeros and three poles are used for each class. r_i in Eq.(6.1) for the class 1 are 0.9945, 0.995 and 0.9985, for the class 2, 0.994, 0.995 and 0.9985, respectively. The pole frequencies are 1.0, 2.0 and 3.0 Hz for the class 1, and 1.5,2.5 and 3.5 Hz for the class 2, respectively. All zeros locate on the unit cycle. Figure 6.2 shows the amplitude response of the class 1 filter. The impulse response is shown in Fig. 6.3.

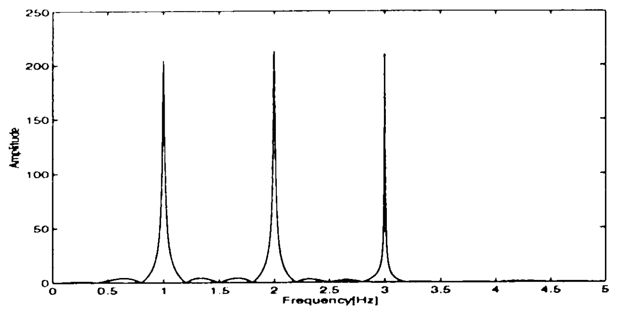


Figure 6.2: Amplitude response of IIR filter to extract class 1 signals.

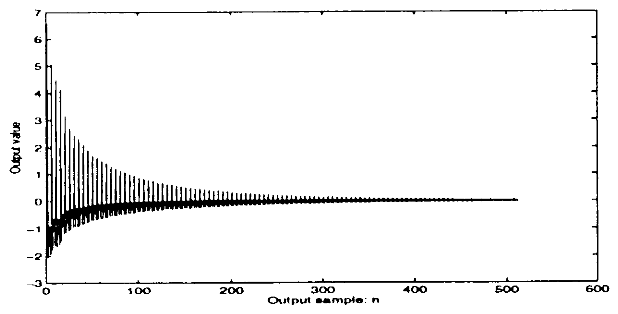


Figure 6.3: Impulse response of class 1 filter.

By using this filter for classification of the multi-frequency signals, the classification rate for the signals with 10 samples and noise free signals is 86.9%. To achieve this accuracy, 2000 output samples are required. This rate is not good compared with that of the FIR filter will be shown in Sec.5.5. The reason is the phase distortion caused by the high-Q amplitude response. By using a lower-Q filter than the above, the classification rate was increased from 86.9% to 95.4%. In this case, r_i in Eq.(6.1) are changed to 0.94, 0.94 and 0.98 for the class 1, and 0.92, 0.935 and

0.98 for the class 2. In the lower-Q IIR filter, 200 output samples are required. On the other hand, since the FIR filter always can guarantee the linear phase, a very high-Q filter can be effectively used as shown in Fig. 6.1.

(4) Maharanobis Generalized Distance (MGD)

As shown in Eq.(5.1), the MGD uses the covariance matrix of the signals that correspond to the training signals for the MLNN. The signals are generated by using Eq.(4.2), then the signals are correlated to each other, so the covariance matrix becomes a singular matrix. To make the covariance matrix be a non-singular matrix, small white noise in the range of ± 0.001 are added to the signals.

The covariance matrix is calculated by using the signals that correspond to the training signals of the MLNN. According to increasing of the number of the signal to calculate the covariance matrix, the accuracy of the covariance of the class region is increased, and the generalization is more effective. In this chapter, 200 signals are used to calculate a covariance matrix for each class. Ineffectiveness of using more than 200 signals is confirmed by further experiment.

(5) Maximum Entropy Method (MEM)

Auto-correlation sequence of the pth class signals are used to get the a_0 and $\{a_k\}$ of the prediction-error filter of the pth class. From [55], the limit of M is $M < (2 \sim 3)\sqrt{(N)}$. From above equation, M is decided as $M = 7$ for 10 sample signals, $M = 10$ for 20 sample signals. They are decided by trials.

6.3.2 Classification Rules

The classifications by LSP methods are carried out as described in Sec.5.2. However, the Fourier transform and the MEM methods use relaxed criterion as follows.

Criterion Define an amplitude response of the input signal at $f_{pr}, p = 1 \sim P, r = 1 \sim R$ as A_{pr}. The input signal is classified into the class that includes the maximum number of A_{pr} is maximum in $A_{p'r}$ at $f_{pr}, r = 1 \sim R$. Where, p' indicates all classes without pth class.

6.4 Computational Complexity

Normalized computational complexity (NCC) is defined to compare classification performance based on the same number of computations. The parameter for each method and the calculation of NCC is described in the followings. The number of samples is N, and the number of class is P. In the NCC, the inner product of two N-dimensional vector is normalized as unity.

(1) Multilayer Neural Network (MLNN)

The NCC for the MLNN is calculated for the network architecture that performs the highest classification for the training and the testing signals. The parameter of the NCC for the MLNN is the number of the hidden units. After the training converges, the hidden unit outputs approach to 1 or 0 [45]. So, the sigmoid function can be replaced by a threshold function in the test phase. Therefore, the calculation of the sigmoid function is omitted from NCC. In this case, $NCC = M + (MP/N)$.

Here, M is the number of the hidden units, and P is the number if the output units.

(2) FIR filter 1 (FIR1)

From Eq.(5.3), one output sample of the FIR1 is calculated by the inner product of the input signal and the filter coefficients. Then NCC for one output sample is unity. When the number of output samples is K, the number of classes is P, then $NCC = KP$. Here, the parameter of NCC for FIR1 is the number of the output sample K.

(3) FIR filter 2 (FIR2)

FIR2 calculates its one output sample as an inner product of the input signal and the filter coefficients. Due to the architecture of the FIR2, only one sample is used. Then, $NCC = 1$ and there is no parameter of NCC.

(4) IIR filter (IIR)

In the case of N_p pole frequencies and N_z zero frequencies are used and $N_z > N_p$, then N_z of the 2nd-order circuits are used. Each circuit includes five inner products of the signal and the filter coefficients, then computation for one filter output is $N_p \times 5 + (N_z - N_p) \times 3 = 2 \times N_p + 3 \times N_z$. Then $NCC = (2 \times N_p + 3 \times N_z)/N$. N_p is as the same as the number of frequencies included in one class. The number of the output samples is the parameter.

(5) Fourier transform (Fourier)

The Fourier transform of the signal is given by Eq.(5.2). In this equation, the inner products of the signal and the complex number of the Fourier kernels are required. The inner product of the complex number is counted as twice of the real number's. If a signal includes R frequency components, then $NCC = 2PR$. The parameter of NCC is the number of observation frequency components R.

(6) Euclidean Distance (Euclid)

If the template and the signal are N-dimensional vectors, and the computation of the Euclidean distance is as the same as the inner product's, then $NCC = PNM$. Here, M is the number of the templates and is the parameter of NCC.

(7) Maharanobis generalized distance (MGD)

To calculate the covariance matrix of the signals that correspond to the training signals of the MLNN, if the number of the signals is M, M inner products are required. However, this covariance matrix can be used to measure the distance, and no re-calculation is required. Then this computation is omitted from NCC. From Eq.(5.1), if the signal is N-dimensional vector, $NCC = (N^2+N)P/N \fallingdotseq NP$. There is no parameter of NCC.

(8) Maximum entropy method (MEM)

The computation of solving the prediction-error filter is omitted from NCC. Because, this process is as the same as the design of the filter and the training phase of the MLNN. From Eq.(5.9), $NCC = 2PMO/N$. M is the number of the observation frequencies, and O is the order of the filter. The parameter is M.

The number of parameters for each method is listed in Table 6.1 and 6.2. Table 6.1 is for limited computation. In the case of the limited computation, the computations of the methods are the same as the computation of the MLNN. The number of parameters is integers, so NCC for all methods is not exactly the same. Table

6.2 is for not limited computation. In this case, the number of parameter is set to achieve the highest accuracy.

Table 6.1: Normalized computational complexities and the number of parameters(Computation is limited).

Methods	N=10		N=20	
	NCC	NP	NCC	NP
MLNN	3.6	3	3.3	3
FIR1	4.0	2	4.0	2
FIR2	2.0	-	2.0	-
IIR	5.1	1	5.1	2
Fourier	4.0	1	4.0	1
Euclid	4.0	2	4.0	2
MGD	–	–	–	–
MEM	2.8	1	4.0	2

N: Number of samples

NP: Number of parameter

As described above, NCC of the FIR2 is the number of the classes P and for the MGD is multiple of the number of the signal samples N and P. These parameters are decided by the classification problem, then NCC is fixed value for these methods. From Tables 6.1 and 6.2, for 10 samples signal and two class classification, NCC is

Table 6.2: Normalized computational complexities and the number of parameters(Computation is not limited).

Methods	N=10		N=20	
	NCC	NP	NCC	NP
MLNN	48	40	44	40
FIR1	20	10	20	10
FIR2	-	-	-	-
IIR	1020	200	510	200
Fourier	12	3	12	3
Euclid	400	200	400	200
MGD	20	-	40	-
MEM	84	3	120	3

N: Number of samples

NP: Number of parameter

2 for the FIR2, and for the MGD, NCC is 20. Under the same condition, NCC for the MLNN is 3.6. Therefore, FIR2 is used when computation is limited, the MGD is used when computation is not limited.

6.5 Multi-frequency Signal Classification

6.5.1 Classification of Multi-frequency Signals

The following two conditions are investigated: the number of computations is limited or not limited. In the former case, computations of the LSP methods are decided as almost the same as in the MLNN method.

The classification rates with limited computations are listed in Table 6.3 in percentage.

In the case of the computation is limited, for the Fourier transform method and for the MEM method, only one observed frequency is used for each class. Then there are ${}_3C_1 \times_3 C_1 = 9$ combinations of the observed frequencies of two classes. So, the classification rates are calculated for nine combinations, and the average of them are listed on this Table. In the Table, the multilayer neural network is denoted as the MLNN, the FIR filter of extracting the specified frequency is denoted as FIR1, FIR filter of suppressing the specified frequency is denoted as FIR2, IIR filter is denoted as IIR, the Fourier transform denoted as Fourier, the pattern matching method using Euclidean distance is denoted as Euclidean, the pattern matching method using the Maharanobis generalized distance is denoted as MGD, and the Maximum entropy method is denoted as MEM, respectively.

From this table, the MLNN method can provide higher performance than the LSP methods. The classification rates of using the signals with 20 samples are better than those of the signals with 10 samples. In the LSP methods, the classification rates are higher for 20 samples' signals than that of 10 samples'. Therefore, non-linearity is notable for 10 samples' signals and is not notable for 20 samples'.

Classification rate of IIR filter for 20 sample signals is worse than that of FIR filter. The main reason of this difference comes from a recurrent structure of IIR filter. If the computation is limited, the output samples in the transient state become dominant in the output power, and accuracy is decreased.

Table 6.3: Probability of exact signal classification in percentage when computation is limited.

Methods	N=10		N=20	
	NFS	NS	NFS	NS
MLNN	97.6	85.4	97.4	90.6
FIR1	4.7	3.7	100	87.5
FIR2	100	50.3	100	51.3
IIR	0.0	0.0	49.0	49.0
Fourier	56.1	53.6	77.9	76.7
Euclid	49.6	52.1	59.4	62.0
MGD	-	-	-	-
MEM	60.8	56.8	87.7	87.3

N : Number of samples

NFS : Noise Free Signal, NS :Noisy Signal

For noise free signals, the FIR2 method provides higher classification rate than the MLNN, however, it is not a case for noisy signal. Because, the FIR2 designed to suppress specified frequency components, however, additive noise is the randomly generated, then the noise spectrum becomes very broad. So, this noise cannot be suppressed.

In the case of the computation being not limited, the classification rates are shown in Table 6.4. The number of the parameters is increased; the number of the hidden units is increased for the MLNN, the number of output samples is increased for FIR1 and IIR, the observation frequency is increased for Fourier and the MEM, and the number of the templates is increased for the Euclidean and the MGD, respectively.

The classification rates of the LSP methods can be improved. They are almost the same in all methods.

For the MLNN method uses the valley shape activation function [25] instead of the sigmoid function in the hidden layer.

6.5.2 Relation Between Computational Complexity and Classification Rates

The relation between the classification rate and the computational complexity is investigated based on NCC. About NCC, refer to Sec.6.4.

Figure 6.4 shows the classification rates of each method with respect to the NCC. This figure obtained by increasing the number of parameters of each method, and

Table 6.4: Probability of exact signal classification in percentage when computation is not limited.

Methods	N=10		N=20	
	NFS	NS	NFS	NS
MLNN	100	90.6	100	99.3
FIR1	100	90.5	100	99.8
FIR2	-	-	-	-
IIR	95.4	86.5	100	99.8
Fourier	70.6	65.7	100	94.8
Euclid	86.0	79.5	100	99.5
MGD	100	90.2	100	99.7
MEM	62.9	63.7	97.3	95.4

N : Number of samples

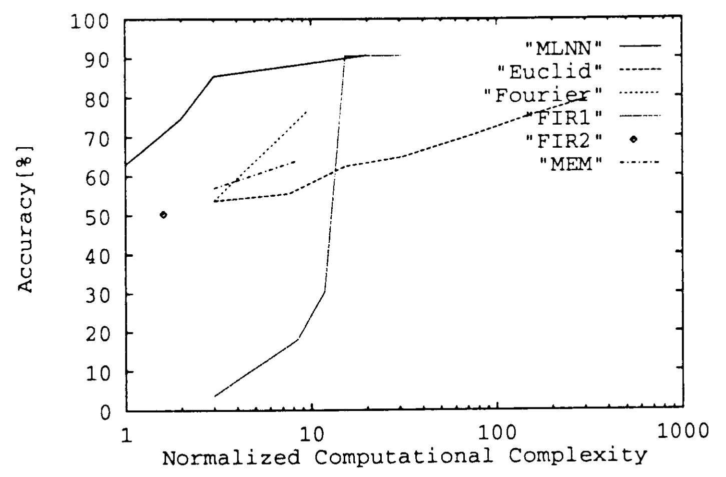
NFS : Noise Free Signal, NS : Noisy Signal

then examined the classification rate. The parameter of each method is explained in Sec.6.4. The noisy signals are used in this investigation. From the figure, for higher NCC, the performances of all the methods are almost the same. However, as the NCC decreases, the LSP methods drastically decrease the classification rates while the MLNN method can still keep relatively high classification rates. Therefore, for all NCC, the MLNN can provide good classification performance.

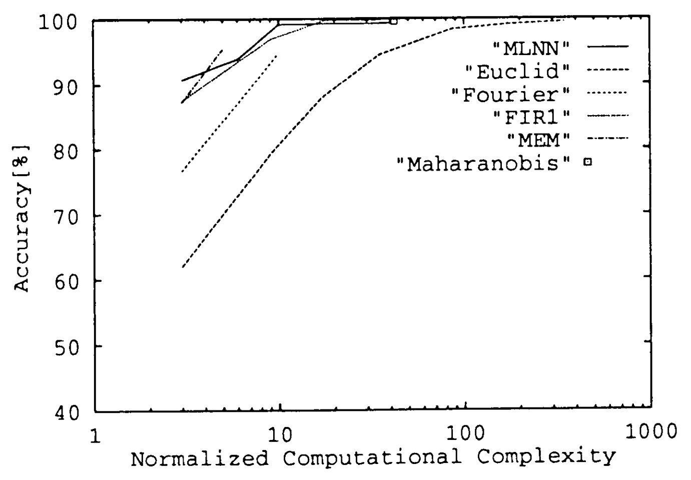
From above investigations, the multi-frequency signal with 10 to 20 samples includes enough information for the classification by LSP methods if they use enough computations. Because, their classification rates are sufficiently high when the computation is not limited. However, this information is not enough when LSP's computations are limited, then the classification rates are drastically decreased. The MLNN keeps a high degree of freedom to form a class region in an N-dimensional space when the circuit scale is small, and this realizes a robustness to the computations.

6.5.3 Learning Ability of Multilayer Neural Network

As discussed in Sec.2.4, when a large number of hidden units are used, it is difficult to converge to the best solution. The initial connection weights should be carefully selected. When random numbers are used as the initial connection weights, the MLNN could not achieve good classification rates as the filters. However, the coefficients of the FIR filter are used as the initial connection weights, the MLNN achieved the same classification rates as the filters'. In this case, the valley shape function is used in the hidden unit. The valley shape function rectifies unit input and it can detect the signal amplitude. Although, this function can be realized by using two sigmoid functions, the former can make fast convergence possible.



(a) 10 samples



(b) 20 samples

Figure 6.4: Classification rates of signal classification by the MLNN method and the linear signal processing methods for (a)10 samples and (b)20 samples noisy signal.

6.5.4 Signal Detection Regions Formed by Multilayer Neural Networks

The signal detection regions formed by the MLNN are investigated based on the hidden layer outputs and the connection weights from the input layer and the hidden layer. From the computer simulation results, the difference of the classification rates between the MLNNs and the LSP methods is clear for ten samples signals' classification. Then training signals of ten samples noisy signals are used in this analysis. The number of the signals is 400 for two classes.

The connection weights values are listed in Table 6.5. In this table, the 1st and the 2nd output units respond to the signal class one and two, respectively. The connection weights from 1st, 2nd and 3rd hidden units and the bias unit to the 1st output unit are -18.95, 18.27, 11.63 and -2.0, respectively. The connection weights to the 2nd output unit are the opposite polarity to those of the 1st one. This symmetry of the connection weights polarity induced by the symmetry of the target signals (1,0) and (0,1).

From these connection weights, the following four patterns are possible to activate the 1st output unit. (Hidden units: 1st, 2nd, 3rd)=(L,H,H), (L,H,L),(H,H,H) and (L,L,H). Where H and L mean high and low level output, respectively.

On the other hand, the following three patterns are available to activate the 2nd output unit, (Hidden unit: 1st, 2nd, 3rd)=(H,L,L), (H,H,L) and (H,L,H). This analysis is farther compared to the actual output patterns.

Table 6.5: Connection weights of hidden layer and output layer

	Output	
Hidden	1st	2nd
1st	-18.95	18.95
2nd	18.27	-18.27
3rd	11.63	-11.63
Bias	-2.0	2.0

Table 6.6 shows the actual hidden layer outputs distribution for the input signals. From this table, for the class one signals, two patterns are obtained out of four patterns given by the analysis above. For class two, all the patterns are obtained. From this results, it is confirmed that the MLNN has higher degree of freedom of forming the signal detection regions, and effectively classifies the multi-frequency signals.

6.5.5 Robustness of MLNN to Noise Level Changes

Robustness for noise level changes is guaranteed by the filters. However, this kind of robustness is not always guaranteed by the MLNN. Then, the robustness of the MLNN for noise level changes is further investigated.

Table 6.6: Hidden unit output distribution.

Class 1				Class 2			
Hidden unit			NS	Hidden unit			NS
1st	2nd	3rd		1st	2nd	3rd	
H	H	H	112	H	H	H	0
H	H	L	0	H	H	L	52
H	L	H	0	H	L	H	109
H	L	L	0	H	L	L	39
L	H	H	0	L	H	H	0
L	H	L	0	L	H	L	0
L	L	H	88	L	L	H	0
L	L	L	0	L	L	L	0

NS: number of signals

Analysis of connection weight

By comparing the Eqs.(2.1) and (2.3), the connection weights between a hidden unit and input layer correspond to the filter coefficients $h_p(n-k)$. Then the connection weights between the input layer and hidden layer are analyzed by using Fourier transform. Figures 6.5 and 6.6 are the amplitude responses of the connection weights trained noisy and noise free signals. The numbers of the input units and the hidden units are ten and three, respectively.

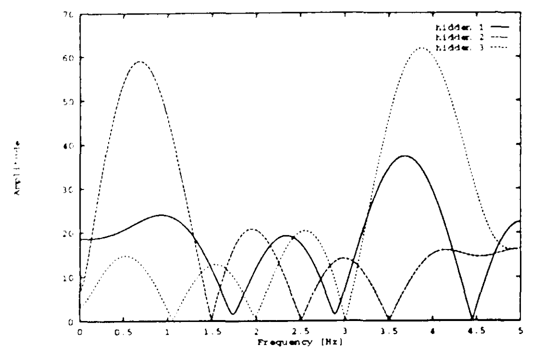


Figure 6.5: Fourier transform of sets of connection weights trained with noise free signals.

Form above two figures, the amplitude response of the connection weights suppressing other class frequencies. When the MLNN is trained using noise free signals, there are two types of amplitude response; one is suppressing class 1 frequencies and the other is suppressing class 2 frequencies. However, when noise signals are used to train the MLNN, only one type of the amplitude response is obtained. From this result, the training by noisy signals is harder than that case noise free signals.

The next figure shows the amplitude responses of Figs.6.5 and 6.6 in the same graph. The connection weights of the input layer and the 3rd hidden unit are used. From this figure, when the MLNN trained by noisy signals, the amplitude response slightly changed into flatter than that of trained using noise free signals. So, the MLNN adapted to the noisy signals by changing its connection weight to have insensitive amplitude response. The FIR filter has a sharp amplitude response and can sufficiently suppresses non-interest frequencies. The MLNN, it does not

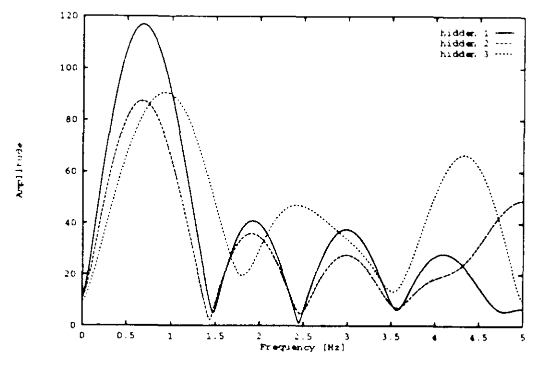


Figure 6.6: Fourier transform of sets of connection weights trained with noisy signals.

have such a sharp amplitude response.

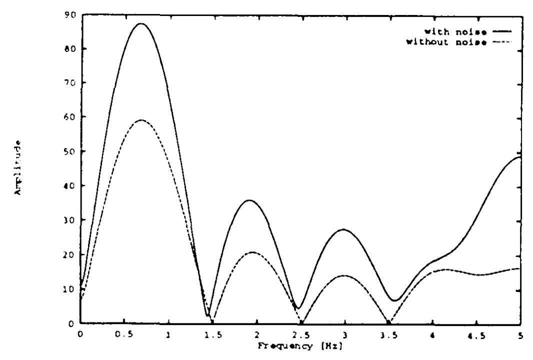


Figure 6.7: Fourier transform of two sets of connection weights trained with noise free and noisy signals.

When the number of hidden units is increased from three to 100, the amplitude response of the connection weights is changed as shown in Fig.6.8. In this case, the amplitude response is similar to that of the FIR filter.

From above analysis, the MLNN achieved amplitude response that can classify the signals. However, the amplitude response of the connection weights is changed

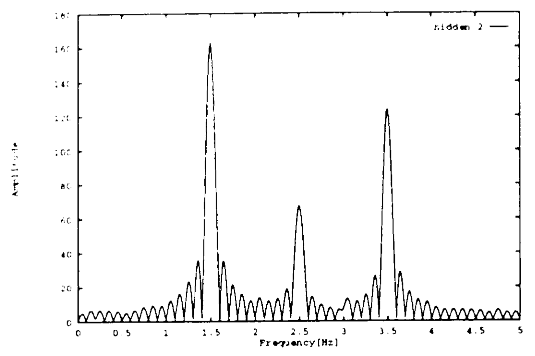


Figure 6.8: Fourier transform of a set of connection weights trained with noise free signals. Number of hidden units is 100.

due to number of hidden units. So, the MLNN can make a suitable amplitude response due to given number of hidden units.

Robustness to white noise

For filter methods, robustness to the noise level change is guaranteed. So, when the noise level is reduced, classification rate will be better. The MLNN is trained with 20 samples including ±0.5 additive random noise. When the noise level is decrease to 0, the classification rate is reduced from 90.6% to 89.7%. In this case, 200 data for each class is used for training. So, generalization for smaller noise signals is not achieved. However, by increasing the number of the training signals from 200 to 400 , the network provides the classification rates of 91.7% for ±0.2 additive noise, and 91.3% for the noise free signals, respectively. Thus, the robustness for noise level change can be guaranteed by training the network with a larger number of the noisy

signals.

6.6 Dial-Tone Recognition

Dial-tone recognition is used to specify a depressed button of the telephone by its tone signal. Dial-tone signal[58] is used in the push button telephone to generate the signals that correspond to the numerical and function buttons. Dial-tone recognition is an application of the multi-frequency signal classification. Two sets of high and low frequencies are used. The low frequency set includes 0.697, 0.770, 0.852 and 0.941 Hz. The high frequency set includes 1.209, 1.366, 1.477 and 1.633Hz. The sampling frequency is 4 Hz. These frequencies are normalized by the sampling frequency as the same as the multi-frequency signal classification.

Sixteen kinds of signal classes are generated by combining the low and high frequencies. One signal class corresponds to one dial-tone of one of the buttons. Table 6.7 shows the combination of the frequencies. From this table, a frequency is included in four dial-tones. Thus, the same frequency is included in the different signal classes. This causes difficulty of signal classification considered the previous subsection. The signal is generated by Eq. (4.2), and the amplitude and phase of the sinusoidal signal are distributed in the same range as the multi-frequency signals. The number of signal samples is 10 or 20.

Table 6.7: Relation between combinations of frequencies in Hz and dial tone classes #1 ~ #16. Frequencies are normalized.

	1.209	1.366	1.477	1.633
0.697	#1	#2	#3	#4
0.770	#5	#6	#7	#8
0.852	#9	#10	#11	#12
0.941	#13	#14	#15	#16

Table 6.8: Classification rates in percentage of dial tone recognition using MLNN method.

Signal Sample	Class. Rate
10	90.6
20	95.7

6.6.1 Classification by Mulitlayer Neural Network

Table 6.8 shows classification rates. Fifty hidden units, whose activation function is the sigmoid function, are used. The classification rates using 20 samples are better than that using 10 samples. In both cases, the classification rates are high. From the results, this complex problem can be solved successively by the MLNN method with a small number of computations.

6.6.2 Classification by FIR Filter

As a useful LSP method, the FIR1 is used to classify the dial-tone signals. Eight kinds of 1000th-order FIR1 filters are designed to extract each frequency component of Table 6.7. A signal FIR1 filter extracts only one frequency component. The FIR1 output powers are calculated, and two of them are added to extract one of the 16 combinations. With computation is limited, the classification rates are compared with those the MLNN method under the same computational complexity. For this purpose, 14 and 10 of the output samples are used to calculate the output power to classify the 10 and 20 samples' signals, respectively. For the case of computation is not limited, two hundreds of the output samples are used to calculate the output samples for the 10 and 20 samples' signals, respectively.

Table 6.9 shows classification rates of the dial-tone recognition. From the table, the classification rates are lower than that of the MLNN method of all the cases. In the case of computation is not limited and using the 20 samples' signals, the performance of the LSP methods is still lower than that of the MLNN method. This result shows that even if using the 20 samples' signals, the frequency resolution is not high enough to achieve the dial-tone classification. From the result of that the MLNN method achieved good classification rates, it can be estimated that non-linearity of this problem is high.

Table 6.9: Classification rates in percentage of dial tone recognition using FIR1 method.

Signal Samples		Output Samples	Class. Rate
10	LT	14	23.3
	NLT	200	41.2
20	LT	10	79.4
	NLT	200	83.6

LT: Computation is limited

NLT: Computation is not limited

6.7 Summary

In this chapter, the classification performance of the MLNNs and the LSP methods have compared based on their classification rates, the number of samples of the signal and computations. From computer simulation results for the multi-frequency signal classification, in the case of the computation is not limited, the MLNNs and the LSP methods are the same classification performance. The short observation period affects the classification performance of the LSP methods. This effect is remarkable when the computation is limited.

Therefore, the analytical results of the chapter 2 and 5 are supported by the results of this chapter. The MLNN has a superior to the LSP methods on the classification performance. Especially, the superiority of the classification performance

of the MLNN is remarkable in the dial-tone recognition. For this kind of the complex problem, the MLNN can achieve good classification performance with small computation.

Chapter 7

CONCLUSIONS

In this thesis, the multilayer neural networks (MLNN) applied to frequency selective classification problem has been studied.

First, the classification mechanism of the MLNN is introduced, and its classification performance is discussed theoretically based on a degree of freedom to divide the pattern space due to the signal class distribution. To verify the degree of freedom, the number of the connection weights which achieve linear separable regions at the input of the output unit is counted out. From the result, the MLNN has had high degree of freedom to form the class region. On the other hand, the MLNN has required training using a set of input and desired outputs. When the degree of freedom of the network parameter is smaller than that required by the problem, the accuracy for trained patterns is low, however, the convergence is fast. In the case of the degree of freedom of the network parameter is large, the accuracy for trained patterns is high, however, the convergence is slow.

Second, minimum number of training data selection methods for generalization and on-line training has been proposed. It has been pointed out that the generalization performance is important subject when the MLNN is applied to the signal processing field, because, the generalization performance of the linear filter methods is always guaranteed, however, it is not guaranteed for the MLNN. One is pairing method and is used Euclidean distance to select the nearest data from other class. The other is pairing and training method, and is select data near the class boundary by using semi-optimal network's connection weights. Two classification problems of two-dimensional are used to verify two methods. From the computer simulation results, the pairing and training method can provide better accuracy than the pairing methods. These can be applied to on-line training.

Third, the classification performance of the linear signal processing (LSP) methods are investigated as a pattern classification. The classification by the LSP methods is analyzed based on the distance from the LSP coefficients to the input signal vectors. In this case, the number of samples of the signal and the computational complexity of the LSP methods has been considered when classification performances of the LSP are investigated. When many samples are used, the signals can be classified with small number of samples of the filter outputs. In this case, signals include frequency component to be extracted by LSP methods are located in narrow space far from the coefficients. However, small number of samples are used, the frequency resolution becomes lower, and more accurate analysis is needed for the classification. Therefore, many output samples are required. The signals locate wide area in the

signal space and some of them are near from the coefficients. In both cases, the classification accuracy can be improved by increasing the number of samples. This is corresponded to increasing the number of hidden units in the MLNN. However, the degree of freedom to select coefficients of the LSP has very small, because, the coefficients are designed to extract specified frequency components.

Finally, the classification performance of the MLNN and the LSP methods are compared in the light of computer simulations results. The frequency selective classification is used for this purpose. The signals are classified based on its frequency components. The comparison is curried out based on classification accuracy, number of signal samples and computational complexity. From computer simulation results for the multi-frequency signal classification, in the case of the computation is not limited, the MLNNs and the LSP methods are the same classification performance. The short observation period affects the classification performance of the LSP methods. This effect is remarkable when the computation is limited. Therefore, the analytical results of above two are supported by the computer simulation results. The MLNN has a superior to the LSP methods on the classification performance. Especially, the superiority of the classification performance of the MLNN is remarkable in the dial-tone recognition. For this kind of complex problem, the MLNN can achieve good classification performance with small computation.

Bibliography

[1] M.L.Minsky and S.A.Papert, *Perceptrons. Expanded Edition*, the MIT press, 1988.

[2] D.E.Rumelhart and J.L.McCelland et al., *Parallel Distributed Processing*, the MIT Press, 1993.

[3] B.Widrow and M.E.Hoff, "Adaptive switching circuits," *IRE WESCON Conv. Rec.*, pt. 4, pp. 96–104, 1960.

[4] S.Haykin, Adaptive filter theory (2nd edition), *Perntice-Hall Inc.*, 1991.

[5] K.Ng and R.P.Lippmann, "A comparative study of the practical characteristics of neural network and conventional pattern classifiers," *Neural information processing systems 3*, pp. 970–976, 1990.

[6] K. Funahashi, "Approximate realization of identity mappings by three-layer neural networks (in Japanese)," *IEICE Trans.*, vol. J73-A, no. 1, January, pp. 139–145, 1990.

[7] S.Amari, "Mathematical foundations of neurocomputing," *Proceedings of IEEE*, vol. 78, No. 9, pp. 1443–1463, Sept. 1990.

[8] B.Widrow and M.Lehr, "30 years of adaptive neural networks: perceptron, madaline, and backpropagation," *Proceedings of IEEE*, vol. 78, No. 9, pp. 1415–1442, Sept. 1990.

[9] E.Levin, N.Tishby and S.Solla, "A statistical approach to learning and generalization in layered neural networks," *Proceedings of IEEE*, vol. 78, No. 9, pp. 1568–1574, Sept. 1990.

[10] T.Poggio and F.Girosi, "Networks for approximation and learning," *Proceedings of IEEE*, vol. 78, No. 9, pp. 1481–1497, Sept. 1990.

[11] Y.Wada and M.Kawato, "Estimation of generalization capability by combination of new information criterion and cross validation(in Japanese)," *IEICE Trans.* vol. J74-D-II, No. 7, pp. 955–965, July, 1991.

[12] V.N.Vapnik and A.Y.Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theoretical Probability and Its Applications*, No. 17, pp.264–280, 1971.

[13] M.Hagiwara, "Back-propagation with artificial selection -Reduction of the number of learning times and that of hidden units-(in Japanese)," *IEICE Trans.* vol. J74-D-II, No.6, pp.812–818, June, 1991.

[14] T.Oshino, J.Ojima, and S.Yamamoto, "Method for gradually reducing a number of hidden units on back propagation learning algorithm," *IEICE Trans.* vol. J76-D-II, No. 7, pp.1414–1424, July, 1993.

[15] K.Nakayama and Y.Kimura, "Optimaization of activation functions in multilayer neural network," *Proceedings of International conference on neural networks*, vol. 1, pp.431–436, June, 1994.

[16] J.Sietsma and R.J.F.Dow, "Neural net pruning – Why and How." *Proceedings of IEEE International Conference of Neural Network*, vol. 1, pp.325–333, 1988.

[17] J.Sietsma and R.J.F.Dow, "Creating artificial neural networks that generalize," *Neural Networks*, vol. 4, pp.67–79, 1991.

[18] J.Zhan and F.Li, "A self-organization neural network and detecting signals," *Proceedings of INNS World congress on neural networks*, Portland, Oregon, vol. IV, pp. 748-751, 1993.

[19] R.Vanderbeek and A.Garper, "A back-propagation network for analog signal separation in high environments," *Proceedings of International joint conference on.neural network*, vol. I, pp. 664–669, Boltimore MD, 1992.

[20] K.A.Al-Mashouq and I.S.Reed, "The use of neural nets to combine equalization with decoding for severe intersymbol interference channels," *IEEE Trans. Neural Network*, vol. 5, No. 6, pp. 982–988, 1994.

[21] D.M.Hummels, W.Ahmed, and M.T.Musavi, "Adaptive detection of small sinusoidal signals in non-gaussian noise using an RBF neural network," *IEEE Trans. Neural Network*, vol. 6, No. 1, pp. 214-219, 1995.

[22] Z.H.Michalopoulou, L.W,.Nolte, and D.Alexandrou, "Performance evaluation of multilayer perceptrons in signal detection and classification," *IEEE Trans. Neural Network*, vol. 6, No. 2, pp. 381–386, 1995.

[23] G.Veciana and A.Zakhor, "Neural net-based continuous phase modulation receivers," *IEEE Trans. communications*, vol.40, No.8, 1992.

[24] D.P.Bouras and D.Makrakis, "Neural- net based receiver structures for single- and multi-amplitude signals in interference channels," *Proceedings of IEEE Workshop on neural networks for signal processing IV*, pp.535–544, 1994.

[25] K.Nakayama and K.Imai, "A neural demodulator for amplitude shift keying signal," *Proceedings of ENNS International conference on artificial neural network*, Sorrent, Italy, vol. 2, pp. 1017-1020, 1994.

[26] K.Ohnishi and K.Nakayama, "A neural demodulator for quadrature amplitude modulation signals", *Proceedings of IEEE International conference on neural networks*, Washington, DC, pp. 1933-1938, June 1996.

[27] K.Hara and K.Nakayama, "High resolution of multi-frequencies using multilayer networks trained by back-propagation algorithm," *Proceedings of INNS World congress on neural networks*, Portland Oregon, vol. IV, pp. 675–678, 1993.

[28] K.Hara and K.Nakayama, "Classification of multi-frequency signals with random noise using multilayer neural networks," *Proceedings of IEEE International Joint Conference on Neural Network*, Nagoya Japan, vol.I, pp.601–604, 1993.

[29] S.Haykin, "Neural networks expand SP's horizons," *IEEE Signal Processing Magazine*, pp. 24–49, March, 1996.

[30] A.C.Tsoi and R.A.Pearson, "Comparison of three classification techniques, CART, C4.5 and multi-layer perceptrons," *Neural Information Processing System 3*, pp. 963–969, 1991.

[31] L.Atlas, R.Cole et al., "Performance comparison between backpropagation networks and classification trees on three real-world applications," *Neural Information Processing System 2*, pp. 622–629, 1990.

[32] S.L.Gish and ,W.E.Blanz, "Comparing the performance of connectionist and statistical classifiers on a image segmentation problem," *Neural Information Processing System 2*, pp. 614–621, 1990.

[33] K.Hara and K.Nakayama, "Comparison of signal classification performance between multilayer neural networks and linear signal processing methods (in Japanese)," *Information Processing Society of Japan Trans.*, vol. 38, No. 2, pp. 245–259, 1997.

[34] K.Hara and K.Nakayama, "Selection of minimum training data for generalization and on-line training by multilayer neural networks," *Proceedings of IEEE*

International conference on neural networks, vol. , pp.436–441, Washington. DC, June, 1996.

[35] K.Hara and K.Nakayama, "Comparison of activation functions in multilayer neural network for pattern classification," *Proceedings of IEEE International conference on neural networks*,Orland, Florida, vol. V, pp. 2997–3002, 1994.

[36] K.Hara and K.Nakayama, "Effects of activation functions in multilayer neural network for noisy pattern classification," *Proceedings of INNS World Congress on Neural Network*, vol. 3, pp.767–772, San Dego, California, June 1994.

[37] K.Hara and K.Nakayama, "Multi-Frequency Signal Classification by Multilayer Neural Networks and Linear Filter Methods," *IEICE trans. Fundamental*, (to be printed)

[38] K.Hara and K.Nakayama, "Signal classification based on frequency analysis using multilayer neural network limited data and computation," *Proceedings of IEEE International Conference on Neural Networks*, vol. 1, pp.600–605, Parth, Australia, Nov. 1995.

[39] T.M.Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electric Computers*, vol. EC–14, pp. 326–334, 1965.

[40] S.Haykin. *Neural Networks – A comprehensive foundation*, pp. 57–59, Macmillan College Publishing Company, 1994.

[41] A.Kowalczyk, "Counting function theorem for multi-layer networks," *Neural Information processing system 6*, pp. 375–382 1995.

[42] J.Makhoul, A.El-Jaroudi, and R.Schwarts, "Partitioning Capabilities of Two-Layer Neural Networks," *IEEE Trans. Signal processing*, vol. 39, No. 6, pp. 1435–1440, 1991.

[43] G.J.Gibson and C.F.N.Cowan, "On the decision regions of multilayer perceptrons," *Proceedings of IEEE*, vol. 78, pp. 1590-1594, 1990.

[44] G.Nakamura, "Four dimensional geometry (In Japanese)," *A separate volume of Mathematical sciences – Dimension –*, Saiensu-sya, pp. 38–43, April 1996.

[45] K.Nakayama, S.Inomata, and Y.Takeuchi: "Reductions in number of bits for digital realization of multilayer neural network (in Japanese)," *IEICE Trans.* , vol. J73-D-II, no. 8, pp. 1336–1345, 1990.

[46] C. Cachin, "Pedagogical pattern selection strategies," *Neural Networks*, vol.7 No.1, pp.175–181, 1994.

[47] M. Kutsuwada, A. Taguchi and Y. Murata, "Fixing the generalization areas of neural networks by iterative learning (In Japanese)," *IEICE Technical Report*, NC93-11, pp.81–87, May 1993.

[48] T. Ueda, K. Takahashi, and S. Mori, "A structural learning for multi-layered neural networks by using fuzzy set – A pruning weights and units – (in Japanese)," *IEICE Trans.*, Vol. J78-D-II, 10, pp. 1479–1490, 1995.

[49] K. Fukumizu and S. Watanabe, "Error Estimation and learning data arrangement for neural networks," *Proceedings of IEEE International conference on neural networks*, Orland, Florida, vol. I, pp. 777–780, 1994.

[50] R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, Inc., 1973.

[51] T. Okuno and H. Kume, Multi-value analysis, *Nikka-giren*, pp. 259–272, 1981.

[52] J. B. MacQueen, "Methods for classification and analysis of multivariate observations," *Proceedings of Symp. Math. Statist. and Prob., 5th*, Berkeley, vol. 1, pp.281–297, 1967.

[53] M. R. Anderson, Cluster Analysis for Applications, *Academic Press*, pp.162–163, 1973.

[54] J. S. Lim and A. V. Oppenheim, Advance Topics in Signal Processing, *Prentice-Hall Inc.*, 1992.

[55] H. Akaike, "Power spectrum estimation through auto-regressive model fitting," *Ann. Inst. Statist. Math.*, Vol. 21, pp. 407–419, 1969.

[56] R. Schumit and R. Franks, Multiple DF signal processing: an experimental system, *IEEE Trans. Antennas and Propagation*, vol.AP–34, pp. 281–290, 1986.

[57] L. R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Prentice-Hall, Inc., (1975).

[58] Y.Matsumoto et al., Signal form for push button dial phone (in Japanese)." *Electrical Communication Laboratories Technical Journal*, Vol. 17, no. 11, pp. 2411–2445, 1968.