

PAPER

A Solution for Imbalanced Training Sets Problem by CombNET-II and Its Application on Fog Forecasting

Anto Satriyo NUGROHO[†], *Student Member*, Susumu KUROYANAGI[†],
and Akira IWATA[†], *Regular Members*

SUMMARY Studies on artificial neural network have been conducted for a long time, and its contribution has been shown in many fields. However, the application of neural networks in the real world domain is still a challenge, since nature does not always provide the required satisfactory conditions. One example is the class size imbalanced condition in which one class is heavily under-represented compared to another class. This condition is often found in the real world domain and presents several difficulties for algorithms that assume the balanced condition of the classes. In this paper, we propose a method for solving problems posed by imbalanced training sets by applying the modified large-scale neural network “CombNET-II”. CombNET-II consists of two types of neural networks. The first type is a one-layer vector quantization neural network to turn the problem into a more balanced condition. The second type consists of several modules of three-layered multilayer perceptron trained by backpropagation for finer classification. CombNET-II combines the two types of neural networks to solve the problem effectively within a reasonable time. The performance is then evaluated by turning the model into a practical application for a fog forecasting problem. Fog forecasting is an imbalanced training sets problem, since the probability of fog appearance in the observation location is very low. Fog events should be predicted every 30 minutes based on the observation of meteorological conditions. Our experiments showed that CombNET-II could achieve a high prediction rate compared to the k -nearest neighbor classifier and the three-layered multilayer perceptron trained with BP. Part of this research was presented in the 1999 Fog Forecasting Contest sponsored by Neurocomputing Technical Group of IEICE, Japan, and CombNET-II achieved the highest accuracy among the participants.

key words: neural network, CombNET-II, self-growing algorithm, imbalanced training sets problem, fog forecasting

1. Introduction

Studies on artificial neural networks have been conducted for a long time and various kinds of architectures have been proposed. However, working on the application of neural networks is still a challenge, since many of the conditions in the real world domain do not always satisfy the requirements of the model. One example is that the real-life condition does not always provide balanced information for every case (class). This condition will create several difficulties for algorithms that assume the balanced condition of the classes. Some examples of real world applications with this kind of

feature are the detection of fraudulent telephone calls, spotting unreliable telecommunications customers, rare medical diagnosis such as the thyroid disease in the UCI repository [1], and travel demand forecasting [2], among others.

Some efforts have been exerted to deal with this problem and several algorithms were proposed [1]-[4]. Japkowicz categorized these attempts into three types [3]:

1. Methods in which the class represented by a small data set becomes over-sampled in order to match the size of the other class
2. Methods in which the class represented by a large data set can be downsized in order to match the size of the other class
3. Methods that ignore one of the two classes, altogether, by using a recognition-based inductive scheme instead of a discrimination-based one

Different from these methods, in this paper we present a new solution by combining competitive learning network and multilayer perceptron. The proposed algorithm is the large-scale neural network “CombNET-II” that has been modified to deal with imbalanced training sets problem. The competitive learning network is the one-layer vector quantization neural network trained by self-growing algorithm, and the multilayer perceptron modules are trained by backpropagation algorithm. The first part is called “stem network” and the second part, “branch network”. The original version of CombNET-II is dedicated to solve problems that involve the classification of a large scale of categories such as Japanese Kanji character recognition [5]. In this study, to deal with imbalanced data sets, we have modified CombNET-II by training the stem network to partition only the dominant class. Each partition is unified with the entire data of the subordinate class to create the training sets for branch network modules. This strategy transfers the problem into a more balanced domain to be solved by the branch network modules. Since the branch network modules are independent of each other, they can be trained simultaneously in several machines, and the required training time can be greatly reduced.

Manuscript received July 10, 2001.

Manuscript revised November 26, 2001.

[†]The authors are with the Department of Electrical and Computer Engineering of Nagoya Institute of Technology

The performance of the proposed solution in the fog forecasting problem is then evaluated. The fog forecasting problem is defined by the prediction of the existence of fog at 30-minutes intervals, in a location where the supercooling-fog event occurs at a rate of approximately 0.3% per year.

This paper is organized as follows: Section 2 provides a brief review of the large-scale neural network ‘‘CombNET-II’’. In Section 3, we explain the proposed algorithm to deal with the imbalanced condition. In Section 4, we discuss the application of the proposed algorithm to the fog forecasting problem. In this section we also make a comparison with three-layered MLP and k -nearest neighbor classifier. In Section 5, we give the conclusion of the discussion in this paper.

2. Self-Growing Algorithm & CombNET-II

CombNET is a large-scale neural network that consists of two parts: the one-layer stem network and branch networks consisting of several three-layered multilayer perceptron modules, as shown in Fig.1. This model is dedicated to solve complex large-scale classification problems effectively with high performance level, such as Kanji character recognition [5], and handwritten character recognition problems [6]. The principle of this model is to simplify the classification problem by automatically dividing the vector space into several sub-spaces according to the statistical distribution of the data. This task is performed by the stem network trained by a vector quantization algorithm. Each sub-space obtained by this algorithm is then presented to a corresponding branch network module. Branch network is trained to generate a classification border for each sub-space. This approach will generate a good classification border for each sub-space, and reduce the number of local minima in training the network.

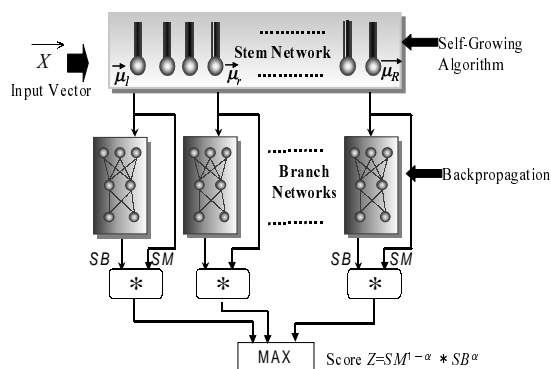


Fig. 1 Structure of CombNET-II

In CombNET-I, stem network is trained by Kohonen’s self-organizing feature map algorithm. However, this algorithm cannot control the size of sub-spaces

generated during the training process. This condition produces different levels of training in branch network, making it difficult to obtain an optimal performance. In CombNET-II, Hotta et al.[5] proposed self-growing algorithm as a training algorithm for stem network instead of SOM. This algorithm functions to make similar sizes of the sub-spaces during the stem network training. The number of sub-spaces is controlled by two parameters: inner potential threshold of a stem neuron (h_{th}) and similarity threshold (r_{th}).

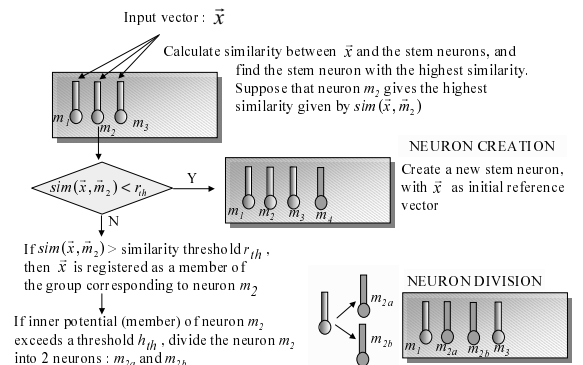


Fig. 2 Self-growing algorithm

Figure 2 shows the outline of the self-growing algorithm. This algorithm is stated as follows:

1. Initialization
One training vector is randomly chosen as the reference vector for the first stem neuron.
2. Data Presentation
Each datum is then presented to all of the stem neurons, and the neuron with the highest similarity is determined. This datum is recorded as a member of a cluster corresponding to the stem neuron.
3. Neuron Creation
If the similarity level of the input datum and the stem neuron is less than the similarity threshold (r_{th}), create a new stem neuron with reference vector as same as the input datum vector.
4. Neuron Division
The inner potential of a stem neuron is defined by the number of members belonging to the cluster corresponding to the stem neuron. If the inner potential of a stem neuron exceeds the inner potential threshold (h_{th}), divide the cluster into two new clusters. The two clusters are separated by a hyperplane that divides the old cluster into two parts with similar inner potential.

After all of the patterns are presented, the first process of this algorithm is completed. The process is

then repeated until there is no significant change of the stem neuron's reference vectors.

Branch networks are three-layered neural networks trained by backpropagation to perform a finer classification for each sub-space. The number of branch network modules is as many as the number of generated stem neurons.

In the recognition phase, a test pattern is presented to all stem neurons, and the distances between test vector x and stem neuron m_r are calculated. $r = 1, 2, \dots, R$, where R denotes the number of generated stem neurons. Few closest stem neurons are selected, and the test vector is presented to the branch networks corresponding to the selected neurons. Final score Z is obtained by:

$$Z = SM^{1-\alpha} \times SB^\alpha, \quad (1)$$

$$0.0 \leq \alpha \leq 1.0.$$

SM stands for score of matching, which is defined by the similarity between test vector and stem neuron.

$$SM = \text{sim}(x, m_r) = \frac{x \cdot m_r}{|x||m_r|} \quad (2)$$

SB stands for score of branch, which is obtained by taking the maximum output value of neurons in an output layer for each branch network. The final result is the classification result of branch network corresponding to the stem neuron m that gives the highest score of Z .

3. Strategy to Deal with Imbalanced Training Sets Problem

To deal with imbalanced training sets problem, we apply self-growing algorithm to perform vector quantization of the dominant class (class that has a larger number of samples). The purpose of this algorithm is to partition the vector space of the dominant class based on the statistical distribution condition. Figure 3 shows the outline of the proposed algorithm.

Let the training set $T = T^{(0)} \cup T^{(1)}$, where each class C^k contains samples as follows:

$$T^{(k)} = \{x_j^{(k)} : j = 1, \dots, n^{(k)}\} \quad \text{for } k = 0, 1, \quad (3)$$

where C^0 is the dominant class, and C^1 is the subordinate class. $n^{(k)}$ represents the number of vectors belonging to class C^k . In the case of imbalanced training sets: $n^{(0)} \gg n^{(1)}$.

The proposed algorithm is as follows:

1. Stem Network is trained to divide input vector space of $T^{(0)}$ into R sub-spaces: $\{T^{(0,r)} : r = 1, \dots, R\}$, where

$$T^{(0)} = T^{(0,1)} \cup T^{(0,2)} \cup \dots \cup T^{(0,R)} \quad (4)$$

and $\{x_i^{(0,r)} : i = 1, \dots, n^{(0,r)}\}$ are the patterns be-

longing to $T^{(0,r)}$. As a result of the training, each sub-space is represented by a stem neuron m_r as a mean vector of the patterns belonging to the sub-space of interest.

$$m_r = \frac{1}{n^{(0,r)}} \sum_{i=1}^{n^{(0,r)}} x_i^{(0,r)} \quad (5)$$

2. Each generated subspace is then unified with a subordinate class.

$$T^{(r)} = T^{(0,r)} \cup T^{(1)} \quad \text{where } r = 1, 2, \dots, R \quad (6)$$

3. $T^{(r)}$ is used to train the branch networks. The number of branch networks is equal to the number of generated stem neurons.

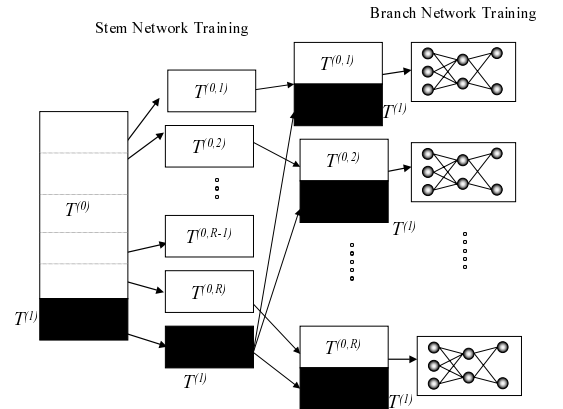


Fig. 3 Proposed algorithm for imbalanced training sets problem

Different from the conventional methods described in Sect.1, the proposed algorithm offers a new strategy for solving the imbalanced training sets problem by combining competitive learning network (stem network) and multilayer perceptron modules (branch networks). Stem network works to simplify the classification task by performing vector quantization to the feature space of the training set, while branch network creates the finer classification border for each sub-space. By applying the algorithm, the classification problem is presented to more balanced domains to be solved by branch networks. Thus, during the training phase of branch networks, the synapses will be updated in a more balanced proportion for both classes to obtain a fine classification border in the feature space. Another advantage of this algorithm is that the branch networks can be trained independently. This approach will reduce the time required to train all modules as long as several machines may be used [7].

4. Application to Fog Forecasting

In this study, the performance of the proposed algorithm was evaluated by applying it to the fog forecasting problem where the probability of a fog event is very low. Moreover, since the trend of the weather may change every year, the generalization of the classifier was evaluated not only by unseen data, but also by unpredicted future trends.

4.1 Fog Forecasting as Imbalanced Training Sets Problem

The database which was used in this experiment was provided by the Forecast Department of the Japan Meteorological Agency [8]. This database consists of 12 years worth of meteorological information from 1984 through 1995, with observations being conducted every 30 minutes. The observations were conducted at the Shin Chitose Meteorological Observatory Station, and the data has been used to support aircraft transportation. The observatory was located at: long. 141.70°E, 42.77°N lat., 25 m above sea level. Note that the observed supercooling-fog events in this location occur at a rate of approximately 0.3% per year, which is the highest among airports in Japan. "Supercooling-fog" is defined as a condition where fog appears while the temperature is below 0°C and the range of visibility is less than 1000 m. The original data were expressed as telegraphic messages, and were converted into numeric expressions by the Numerical Forecasting Section of the Japan Meteorological Agency. The observation included 26 features as shown in Tab.1.

Table 1 Meteorological Information Provided in the Database

No.	Observed feature
1	Year
2	Month
3	Date
4	Time
5	Atmospheric Pressure [hPa]
6	Temperature [°C]
7	Dew Point Temperature [°C]
8	Wind Direction [°]
9	Wind Speed [m/s]
10	Maximum Instantaneous Wind Speed [m/s]
11	Change of Wind (1) [°]
12	Change of Wind (2) [°]
13	Range of Visibility [m]
14	Weather
15	Cloudiness (1st layer)
16	Cloud Shape (1st layer)
17	Cloud Height (1st layer)
18	Cloudiness (2nd layer)
19	Cloud Shape (2nd layer)
20	Cloud Height (2nd layer)
21	Cloudiness (3rd layer)
22	Cloud Shape (3rd layer)
23	Cloud Height (3rd layer)
24	Cloudiness (4th layer)
25	Cloud Shape (4th layer)
26	Cloud Height (4th layer)

Table 2 Ratio of Number of Patterns of Each Class

Year	Number of Fog Events	Number of No-Fog Events	Ratio
1984	467	16961	1:36
1985	426	17033	1:40
1986	314	17130	1:55
1987	275	17172	1:62
1988	282	17260	1:61
1990	220	17199	1:78
1991	220	17272	1:79
1992	389	17163	1:44
1993	211	17301	1:82
1994	298	17211	1:58

Below is one example of the data:

1984 1 1 0.5 1018 -2.0 -3.0 -1 -1 -1 -1 -1 30 26 0 2 7 0 4
25 -1 -1 -1 -1 -1 -1

The data consist of 26 features corresponding to information given in Tab.1. These features indicate that the observation was made on January 1, 1984 at 0:30 am. At this point, the atmospheric pressure is 1018 hPa, and temperature is -2.0°C. There are many -1 values in the data and these are regarded as dummy values. The database defined this value, because there was not any observation for the feature. For temperature, the dummy was expressed as 999 to avoid ambiguity with -1°C.

The database was then classified into two classes: "fog event" and "no-fog event" depending on the values of the "range of visibility" and the "weather". In the database, fog event is defined as the condition when the "range of visibility" is less than 1000 m and "weather" shows the appearance of fog. In this database, the "weather" feature that shows the appearance of fog is expressed as an integer between 40 and 49. The classification results for all years are shown in Tab.2.

Table 2 shows that the ratio of the two classes is extremely imbalanced, which leads to a difficult condition when backpropagation-trained neural networks are used. To deal with this imbalanced condition, we proposed a solution by the use of the modified CombNET-II.

4.2 Creating Training Sets

Selection of the database should be based on the order of the year. Therefore, the training set should be chosen from the previous years while the test set, from the later ones. In this experiment, the training set comprised data from years 1988, 1990, 1991, 1992, and 1993 and the performance was evaluated by predicting fog events of the next year i.e., 1994 data. This data selection implied that the generalization of the classifier could be evaluated not only by unseen data, but also by the unpredicted weather trends of 1994.

In this experiment, we dropped "year", "month", "date", "time", "range of visibility", "weather" (as required in the fog forecasting contest, prediction should be carried out based on meteorological information

where “range of visibility” and “weather” features are excluded).

The database was then divided into two classes: fog event class and no-fog event class based on the value of range of visibility and weather for each pattern. After the division, we obtained a training set with the numbers of fog event patterns and no-fog event patterns being 1322 and 86195, respectively.

4.3 Data Normalization

Since the database contained many dummies that were expressed as -1, normalization to the interval value of [-1,+1] was an appropriate choice while the dummies were converted into 0.

The data should be normalized appropriately based on the physical meaning of each feature. We denote the meteorological information by ξ_i , where $1 \leq i \leq 26$, and the normalized data by ϕ_j , where $1 \leq j \leq J$. J is the dimension of the input vector presented to the classifier. Since “year” (ξ_1), “month” (ξ_2), “date” (ξ_3) and “time” (ξ_4) were dropped, ϕ_1 became the normalized value of atmospheric pressure (ξ_5). Range of visibility (ξ_{13}) and weather (ξ_{14}) features were not used to create the input vector.

Part of the meteorological information was provided as an angle ($\xi_8, \xi_{11}, \xi_{12}$) with the interval value of [0,360]. These data were normalized as two-dimensional vectors that showed the location of the points in a circle. Wind direction (ξ_8) was normalized by the following equation:

$$\phi_4 = \sin\left(\pi \frac{\xi_8}{180}\right), \tag{7}$$

$$\phi_5 = \cos\left(\pi \frac{\xi_8}{180}\right). \tag{8}$$

The same operation was applied for ξ_{11} (change of wind (1)) , and ξ_{12} (change of wind (2)).

A specific normalization was also applied for cloud shape in the 1st (ξ_{16}), 2nd (ξ_{19}), 3rd (ξ_{22}) and 4th (ξ_{25}) layers. These features were provided as integer values from 1 to 11, and each of them showed a different shape of cloud. Table 3 shows the meaning of each value.

Table 3 Cloud Shape

Value	Definition	Value	Definition
-1	No observation	6	Alto cumulus
1	Cumulus	7	Altostratus
2	Stratus	8	Nimbostratus
3	Stratocumulus	9	Cirrus
4	Cumulonimbus	10	Cirrostratus
5	Towering Cumulus	11	Cirrocumulus

Since the values do not show any order, they cannot be expressed as a one-dimensional vector. In this experiment, *1-of-c coding* [9] was used to express the cloud shape as an 11-dimensional vector. We denote the normalized vector for cloud shape in the 1st layer (ξ_{16}) by ϕ_{12+i} , where $i = 1, \dots, 11$. The value of ϕ_{12+i} is given as follows:

$$\phi_{12+i} = \begin{cases} -1 & \text{if } i \neq \xi_{16} \text{ and } \xi_{16} \neq -1 \\ +1 & \text{if } i = \xi_{16} \\ 0 & \text{if } \xi_{16} = -1. \end{cases} \tag{9}$$

The same normalization was also applied for cloud shape in the 2nd (ξ_{19}), 3rd (ξ_{22}) and 4th (ξ_{25}) layers. For the other features, linear normalizations were applied to transfer the value to the range [-1,+1].

As a result, each pattern was expressed as a 63-dimensional normalized vector where the element had a value at interval [-1,+1]. Table 4 shows the detailed definitions of vector elements. This feature vector was used to train the neural network.

Table 4 Definition of Each Element of the Feature Vector

i	Definition of ϕ_i
1	Atmospheric Pressure
2	Temperature
3	Dew Point Temperature
4,5	Wind Direction
6	Wind Speed
7	Maximum Instantaneous Wind Speed
8,9	Change of Wind (1)
10,11	Change of Wind (2)
12	Cloudiness (1st layer)
13,14, ..., 23	Cloud Shape (1st layer)
24	Cloud Height (1st layer)
25	Cloudiness (2nd layer)
26,27, ..., 36	Cloud Shape (2nd layer)
37	Cloud Height (2nd layer)
38	Cloudiness (3rd layer)
39,40, ..., 49	Cloud Shape (3rd layer)
50	Cloud Height (3rd layer)
51	Cloudiness (4th layer)
52,53, ..., 62	Cloud Shape (4th layer)
63	Cloud Height (4th layer)

4.4 Training the Neural Network

In this study, the performance was evaluated based on the total number of correctly predicted events for both classes based on the criteria defined in the 1999 fog forecasting contest [8]. Therefore, the parameters were chosen to minimize the total number of mispredicted events.

Data of “no-fog event” were presented to stem network. Stem network applied self-growing algorithm [5], which performed vector quantization to the vector space of the dominant class. The number of sub-spaces was controlled by adjusting two parameters of this algorithm, i.e., inner potential threshold h_{th} and similarity r_{th} . In this experiment, h_{th} was set at 6610 while r_{th} was 0.4; then, the dominant class was partitioned into 22 spaces. The number of patterns belonging to each cluster is shown in Fig.4, where the horizontal axis represents stem neuron number corresponding to the cluster of interest. For some stem neurons, the number of patterns exceeds the inner potential threshold h_{th} . This is because in our experiments, we did not divide the stem neurons in the second process of the self-growing algorithm to avoid the creation of neurons with relatively small number of patterns.

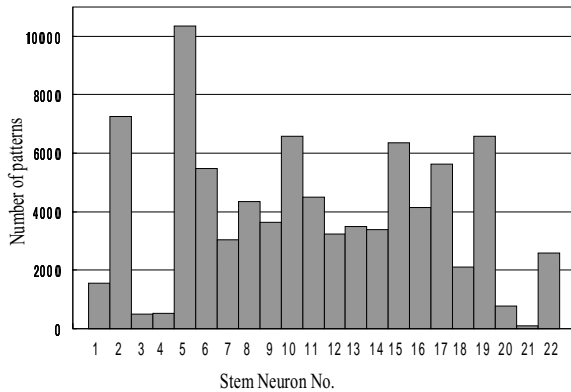


Fig. 4 Number of patterns belonging to each cluster

Each partition was then unified with the “fog event class”, and the class with a smaller number of members was duplicated to match the size of the other class. The data were then presented to branch network. The numbers of neurons for input layer, hidden layer and output layer of branch networks were 63, 100 and 2, respectively.

The training phase was performed in a UNIX environment, using an Athlon 1.2 GHz CPU, Solaris 8 OS machine. Training the stem network needed only about 20 seconds, while to train all the branch modules at a maximum of 1000 iterations, we needed 15 hours and 28 minutes. The average time necessary for one branch module was 42 minutes, while the maximum time for one branch module was 1 hour and 52 minutes. One advantage of CombNET-II is that the decomposition of the task to several branches permits us to use several machines simultaneously, thereby improving time efficiency. This feature permitted us to reduce the necessary time up to around 1 hour and 52 minutes if 22 machines were used to train the modules independently. This advantage leads us to conclude that the model is appropriate for classification problems involving large amounts of data which are often found in practical applications. After the training phase was finished, the average of MSE for each neuron of branch network’s output layer (CombNET-II) was 3.7×10^{-3} .

4.5 Results & Discussion

We also attempted to solve the same problem using a complete storage type k -nearest neighbor classifier, where all data of the training set were used as prototypes. The k -nearest neighbor classifier is a good choice for performance comparison and is expected to produce a high prediction rate. The result is also compared with backpropagation-trained three-layered MLP which assumes the same structure as branch network.

Figure 5 shows the correct prediction rate achieved by CombNET-II at several values of α for 1988 data. The result shows that the correct prediction rate of “fog

event” class increases as the value of α increases, while that of “no-fog event” shows the opposite trend. The parameter α was chosen by taking the value that gave the best performance of the classifier. Thus, $\alpha = 0$ was chosen since it gave the smallest number of incorrect predictions. This value implied that the forecasting result was the classification result of branch network corresponding to the stem neuron with the largest similarity to the test vector.

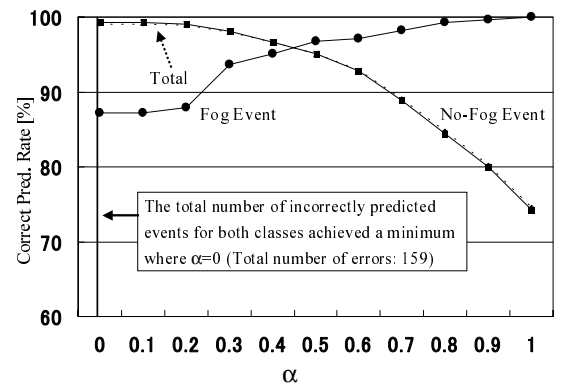


Fig. 5 Correct prediction rate for each class by CombNET-II

The performance of CombNET-II depends on the classification result of stem network while the self-growing algorithm depends on the order of data presentation. Thus, we conducted several experiments by changing the order of data presentation. Training data were divided into four parts; thus, we had 24 combinations of data order for 24 experiments.

The correct prediction rate (CPR) is denoted by CPR_t and the statistical results are shown in Tab.5. Figures inside parentheses show the number of misclassified patterns.

Table 5 Correct Prediction Rate of CombNET-II (test data: 1994)

CPR_t	Total	No Fog	Fog
Max (total)	99.1 % (160)	99.4 % (96)	78.5 % (64)
Min (total)	98.6 % (237)	98.9 % (193)	85.2 % (44)
Ave. (total)	98.9 % (184)	99.3 % (121)	78.9 % (63)

Table 5 shows that CombNET-II achieved a high prediction rate. The best result is achieved by correctly classifying 99.1% of the test data, while the average correct prediction rate is 98.9%. The time needed to classify the test data (data of year 1994) was 11 seconds. Since the number of test patterns is 17509, the average classification time for one pattern is 0.6 ms.

First, the proposed algorithm is compared to three-layered MLP that was trained using an over-sampling scheme. The training set for MLP was created by over-sampling the data of a subordinate class (“fog event”) in order to match the size of the dominant class (“no-

fog event”). MLP had the same structure as those of the branch networks and was trained using backpropagation algorithm. The numbers of neurons for input layer, hidden layer and output layer were 63, 100, and 2, respectively. The two neurons of the output layer corresponded to the dominant class and the subordinate class, and used the pair {0.1,0.9} as the target values. Target value 0.9 was assigned to the output neuron corresponding to the class of input pattern, while 0.1 was the target value for the other one. The network was trained for a maximum of 1000 epochs or until the MSE error was below 10^{-4} .

After the training phase was completed, the average of MSE for each neuron of the output layer of MLP was 9×10^{-3} . Comparing the error level with that of CombNET-II reveals that the vector quantization approach of CombNET-II improved the performance of the network to achieve a lower error level than MLP.

Table 6 Correct Prediction Rate of MLP (test data: 1994)

Total	Dominant Class	Subordinate Class
97.7 % (408)	97.8 % (381)	90.9 % (27)

The classification result of the MLP is presented in Tab.6. Figures inside parentheses show the number of misclassified patterns. Comparison of MLP results with those of the proposed algorithm (Tab.5) shows that CombNET-II outperforms the over-sampling scheme MLP. Although MLP shows better classification results for the “fog event” class, the total correct prediction rate is significantly lower than that of CombNET-II. This is because of the over-sampling approach that was used to create a training set for the classifier, leading the network to overlearn the subordinate class. Since the data of the subordinate class were over-sampled, during the training phase, the synapses of the network were updated more frequently to minimize the error of the output neuron corresponding to the subordinate class, rather than that corresponding to the dominant class. This situation made it difficult for the network to obtain a high total correct prediction rate, since many of the patterns belong to the dominant class could not be correctly classified.

Another disadvantage of the over-sampling scheme MLP is the time required to train the network, which may increase if the model is used to deal with large data sets. In our experiment, the time required to train the over-sampling scheme MLP was 25 hours and 50 minutes, which was 14 times longer than the maximum time required by the branch modules of CombNET-II. If the branch modules of CombNET-II were trained in 22 machines, which is as many as the number of stem neurons, the proposed algorithm would be trained 14 times faster than MLP.

From these results, we concluded that the performance of CombNET-II is superior to that of MLP, and that CombNET-II is a good approach for practical ap-

plications since it can be trained in a much shorter time than MLP.

The second experiment compares the performance of CombNET-II to that of the k -nearest neighbor classifier. The k -nearest neighbor classifier is a nonparametric classifier based on the concept that samples that are close in feature space likely belong to the same class. Although the algorithm of the k -nearest neighbor classifier is simple, it offers sub-optimal results compared to the Bayes decision [10][11][12][13]. In this experiment, we used the *complete storage type* k -nearest neighbor classifier. This approach means that all data of the training set are used as prototypes, and hence a high prediction rate is expected.

In order to deal with an imbalanced data set, the algorithm is modified as follows:

Let us denote the number of subordinate prototypes among the k prototypes by ν_1 , the number of dominant prototypes among the k by ν_0 , the dominant class by C^0 , the subordinate class by C^1 , and the classification result for test pattern x by C_x .

Based on the Euclidean distance, we take k prototypes that are the closest to the test pattern. The class decision is given by:

$$C_x = \begin{cases} C^0 & \text{if } \nu_0 > \nu_1 \times c \\ C^1 & \text{if } \nu_0 < \nu_1 \times c \\ \text{rejected} & \text{if } \nu_0 = \nu_1 \times c, \end{cases} \quad (10)$$

where c is a certain coefficient ($c \geq 1$).

In this experiment, we used several values of k ($k = 1, 2, \dots, 10$). By optimizing parameter k , this algorithm is expected to give a good approximation of the optimal prediction rate.

Figure 6 shows the correlation between the number of misclassified patterns and coefficient c for 9-nearest neighbor classifier ($k = 9$). This result shows that a large value of c improved the prediction rate of the subordinate class. However, choosing a large value of c conversely lowered the prediction rate of the dominant class, and subsequently the total performance decreased. This is because the classifier produced higher scores for the subordinate class patterns if they were among the selected prototypes for the test pattern of the dominant class. Thus, $c = 1.0$ was chosen for $k = 9$, since the classifier produced the smallest total number of incorrect predictions. The same operations were conducted for $k = 1, 2, \dots, 10$ and the results are presented in Fig.7 and Tab.7.

The best result was achieved for $k = 9$ and $c = 1.0$ by correctly classifying 99.1% of test data. Table 7 and Tab.5 clearly show that k -nearest neighbor classifier and CombNET-II achieved similar level of classification. Since k -nearest neighbor classifier is a good approximation of the optimal classification, the results lead us to conclude that CombNET-II achieved similar performance to that of the sub-optimal classifier. Although they have similar performance, CombNET-

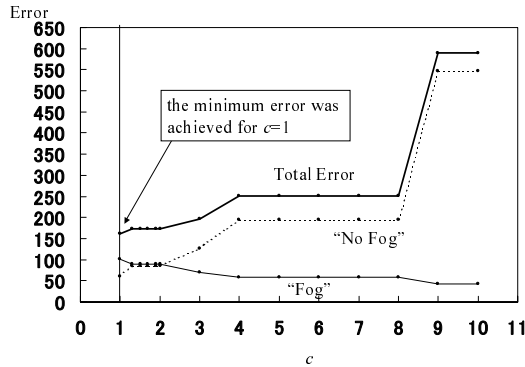


Fig. 6 Number of error predictions for each value of coefficient c of 9-nearest neighbor classifier (test data: 1994)

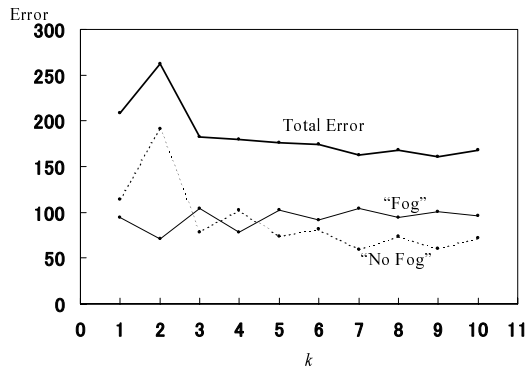


Fig. 7 Number of error predictions for each value k of k -nearest neighbor classifier (test data: 1994)

Table 7 Correct Prediction Rate and Error Number for Each Value k of k -Nearest Neighbor Classifier (test data: 1994)

k	c	Total	Dominant Class (No Fog)	Subordinate Class (Fog)
1	1.0	98.8 % (208)	99.3 % (114)	68.4 % (94)
2	1.1	98.5 % (262)	98.9 % (191)	76.2 % (71)
3	1.0	99.0 % (182)	99.5 % (78)	65.1 % (104)
4	1.1	99.0 % (180)	99.4 % (102)	73.8 % (78)
5	1.0	99.0 % (176)	99.6 % (74)	65.8 % (102)
6	1.1	99.0 % (174)	99.5 % (82)	69.1 % (92)
7	1.0	99.1 % (163)	99.6 % (59)	65.1 % (104)
8	1.1	99.0 % (168)	99.6 % (74)	68.4 % (94)
9	1.0	99.1 % (161)	99.6 % (60)	66.1 % (101)
10	1.1	99.0 % (168)	99.6 % (72)	67.8 % (96)

II has many significant advantages for practical application over the k -nearest neighbor classifier. This is clearly shown by the comparison of the time required for the classification process. In the experiment, to classify all the test patterns, the classification process of the k -nearest neighbor classifier required 35 minutes and 48 seconds. Thus, the average time to classify one pattern is 122.7 ms, which is 204 times slower than that

of CombNET-II. This is because of such disadvantages of k -nearest neighbor classifier as the necessity of storing all of the prototypes and the computational cost incurred in searching them to find the nearest neighbor patterns. In particular, when a large-scale data set is used for the experiment, the implementation of k -nearest neighbor classifier becomes expensive, because of the necessity to store all of the prototypes. Table 8 shows the comparison of memory size for the classification systems of CombNET-II and k -nearest neighbor classifier. Figures inside parentheses show the ratio. This comparison shows that to achieve a similar level of classification, CombNET-II requires much smaller memory than k -nearest neighbor classifier. These features bring many advantages when the proposed model is used for practical applications.

Table 8 Size Comparison of CombNET-II and k -Nearest Neighbor Classifier

CombNET-II	k -nearest neighbor classifier
146630	5513571
(No. of synapses)	(No. of prototypes \times dim.)
(1)	(38.0)

We analyzed the correct prediction rate of “fog event class” of CombNET-II and k -nearest neighbor classifier, and found that CombNET-II achieved a higher rate than k -nearest neighbor classifier. In the case of year 1994 data, the correct prediction rate of “fog event” class for CombNET-II was 78.5% (64 events were not correctly predicted), while that for 9-nearest neighbor classifier was 66.1% (101 events were not correctly predicted). From these misclassified patterns, 63 patterns could not be classified by both CombNET-II and k -nearest neighbor classifier, 38 patterns were correctly classified by CombNET-II but not by k -nearest neighbor classifier, and only one pattern was correctly classified by nearest neighbor but not classified by CombNET-II. This result shows that the prediction performed nonlinearly in CombNET-II was successful for patterns of the subordinate class which were very close in feature space to the dominant class. The k -nearest neighbor classifier could not avoid misclassification of such patterns, since the distance calculations were linearly performed. This result also shows that fog event forecasting is a very difficult problem, since k -nearest neighbor classifier failed to do fine prediction even if the entire data of training set were used as templates. Since predicting the fog event (subordinate class) is very important, the results lead us to conclude that CombNET-II offers a better solution than k -nearest neighbor classifier.

In practical applications, predicting the appearance of fog in the future is important. Therefore, the performance of the proposed algorithm was also evaluated by presenting meteorological observation at a certain time t , and the model should predict the fog ap-

pearance at $t + \Delta t$. As described in Sect.4.1, the interval between one meteorological observation and the next observation is 30 minutes. Thus, the performance of the model was evaluated by training the network to forecast the fog event at $t + \Delta t$ with $\Delta t = 30, 60$ and 90 . In the classification stage, a test vector created from observation of meteorological condition was presented to the classifier; thus, the model should predict the fog appearance in the next 30 minutes, 60 minutes and 90 minutes.

Let us denote the correct prediction rate by CPR_t , and the next types of predictions are denoted by CPR_{t+30} , CPR_{t+60} , CPR_{t+90} , respectively. Data of year 1988, 1990, 1991, 1992 and 1993 were used as training set while those of 1994 were used for performance evaluation. The parameters of CombNET-II, MLP and k -nearest neighbor classifier were the same as those of previous experiments. The order of data presentation for CombNET-II experiments was also changed 24 times for each experiment to obtain the statistics of the results.

The results obtained from CombNET-II are compared to those of k -nearest neighbor classifier and MLP, and are presented in Tab.9 (CPR_t of the classifiers are not presented here, since they have been presented in previous tables). Figures inside parentheses are the number of incorrect predictions. Note that the prediction rate is expressed as the ratio of the number of correct predictions for all classes to the entire data.

Table 9 Correct Prediction Rate of All Methods

Method	CPR_{t+30}	CPR_{t+60}	CPR_{t+90}
k -NN (best)	98.8 % (215)	98.4 % (272)	98.3 % (294)
MLP	94.7 % (918)	93.5 % (1138)	90.4 % (1679)
CombNET-II (Max)	98.8 % (213)	98.6 % (245)	98.4 % (284)
CombNET-II (Min)	97.8 % (378)	97.6 % (422)	97.4 % (461)
CombNET-II (Ave)	98.4 % (271)	98.2 % (320)	97.9 % (368)

The results show that CombNET-II achieves better performance than MLP for all types of predictions. When they are compared to k -nearest neighbor classifier, they show similar levels of prediction rate. Comparing the accuracy of the four types of prediction, we find that the accuracy decreased when the time interval (Δt) with the predicted condition became longer. The difference between CPR_t and CPR_{t+90} obtained by CombNET-II is 0.7% (124 errors), while that by MLP is 7.3% (1271 errors). This result shows that the proposed algorithm gives more stable accuracy than the oversampling approach used in MLP.

In this study, we do not discuss the feature extraction method in the preprocessing for selecting features that have high correlation with the classification task.

This is because the objective of this study is to present a general strategy for dealing with imbalanced data sets, which is also applicable for the other classification problems. Consequently, the training vectors in this experiment might contain features that were actually not necessary for the prediction, and conversely could make the generalization poor. An appropriate feature extraction method will improve the performance of the system; this is an interesting topic for future work.

4.6 Fog Forecasting Contest

Part of this research was presented in the 1999 Fog Forecasting Contest sponsored by the Neurocomputing Technical Group of IEICE, Japan. In this event, the participants were requested to forecast fog events for years 1989 and 1995 using data of the other 10 years. The data for all years except 1989 and 1995 were completely given including all 26 features in Tab.1. For data of years 1989 and 1995, only 24 features were given by removing "Range of Visibility" and "Weather". Therefore, only 10 years of database could be used as training sets and performance evaluation sets. As a result, CombNET-II achieved the highest score among the participants. For details of this contest, please see [14].

5. Conclusion

In this paper, we propose a solution for the imbalanced training sets classification problem by applying the modified large-scale neural network CombNET-II. This approach offers a new strategy for imbalanced data sets by performing vector quantization for dominant class which is implemented by stem network of CombNET-II. In this phase, the training data are decomposed into several groups to simplify the classification task. This decomposition allows us to train each module independently in the second phase, and thus the time needed to train the model can be reduced. This feature makes this model suitable for real world applications which often involve large data sets.

This model was implemented in the fog forecasting problem, where it had to predict the appearance of fog based on meteorological conditions. The prediction should be conducted every 30 minutes in a location where the probability of fog event is very low. A five-year database was used to train the model. The time required for training CombNET-II could become 14 times shorter than the time required for three-layered MLP. Performance evaluation of CombNET-II revealed that it is superior to MLP and achieves a similar level of accuracy to k -nearest neighbor classifier. Analysis of the results shows that CombNET-II produced a higher prediction rate of fog event than k -nearest neighbor classifier. CombNET-II also has such advantages over k -nearest neighbor classifier as the classification time and the required memory size. These results lead us to conclude that CombNET-II offers a better solution for practical applications than k -nearest neighbor classifier and MLP.

Part of this research was presented in the 1999 Fog Forecasting Contest sponsored by the Neurocomputing Technical Group of IEICE Japan as a part of the 1999 IEICE General Conference. In this contest, CombNET-II achieved the best result among the participants.

6. Acknowledgements

The authors wish to express their appreciation to the Forecast Department of the Japan Meteorological Agency for providing meteorological database. The first author is supported by a scholarship from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

- [1] M. Kubat, and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection", In Proc. of the 14th International Conference on Machine Learning, pp.179-186, 1997.
- [2] A.S. Dantas, K. Yamamoto, M.V. Lamar, and Y. Yamashita, "Neural Network for Travel Demand Forecast Using GIS and Remote Sensing", In Proc. of IJCNN2000, vol.4, pp.435-440, 2000.
- [3] N. Japkowicz, "Learning from Imbalanced Data Sets: A Comparison of Various Strategies", International Workshop on Learning from Imbalanced Data Sets, Technical Report WS-00-05, July 2000.
- [4] N. Japkowicz, "The Class Imbalance Problem: Significance and Strategies", In Proc. of the 2000 International Conference on Artificial Intelligence, vol.1, pp.111-117.
- [5] K. Hotta, A. Iwata, H. Matsuo, and N. Suzumura, "Large Scale Neural Network "CombNET-II"", IEICE Trans. Inf. & Syst., vol.J75-D-II, no.3, pp.545-553, 1992 (in Japanese).
- [6] H. Kawajiri, T. Yoshikawa, J. Tanaka, S.N. Anto, and A. Iwata, 21-2721-27 "Hand-written Numeric Character Recognition for Facsimile Auto-dialing by Large Scale Neural Network "CombNET-II"", In Proc. of the 4th EANN, Gibraltar, pp.40-46, June 1998.
- [7] S.N. Anto, S. Kuroyanagi, and A. Iwata, "Fog Forecasting by "CombNET-II"", In Proc. of 1999 IEICE General Conference, pp.323-324, 1999 (in Japanese).
- [8] Explanation about meteorological database which were used in this experiment can be found at http://www.bpe.es.osaka-u.ac.jp/~shouno/IEICE_NC/ (in Japanese)
- [9] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
- [10] R.O. Duda, and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.
- [11] T.M. Cover, and P.E. Hart "Nearest Neighbor Pattern Classification", IEEE Trans. on Information Theory, vol.IT-13, No.1, Jan.1967, pp.21-27.
- [12] K. Fukunaga, and D.M. Hummels, "Bias of Nearest Neighbor Error Estimates", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.PAMI-9, No.1, Jan.1987, pp.103-112.
- [13] K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press, 1990.
- [14] 1999 Fog Forecasting Contest results can be seen at http://www.bpe.es.osaka-u.ac.jp/~shouno/IEICE_NC/result.html (in Japanese)

Anto Satriyo Nugroho received his B.E.degree and M.E.degree in 1995 and 2000, respectively, in Electrical and Computer Engineering from Nagoya Institute of Technology. From 1995 until 1997 he worked for BPPT (Agency for Assessment and Application of Technology) of the Republic of Indonesia. Currently he is a doctoral course student at the Department of Electrical and Computer Engineering, Nagoya Institute of Technology. His research interest is in the field of neurocomputing and applications of neural network.

His research interest is in the field of neurocomputing and applications of neural network.

Susumu Kuroyanagi is a research associate in the Department of Electrical and Computer Engineering, Nagoya Institute of Technology. He received his B.E. degree in 1991, M.E. degree in 1993, and Ph.D. in 1996, all in Electrical and Computer Engineering from Nagoya Institute of Technology. His research interests include neurocomputing, application of pulsed neural networks, and modeling of the auditory nerve system. He is a

member of the Acoustical Society of Japan and the Japanese Neural Network Society.

Akira Iwata is a professor in the Department of Electrical and Computer Engineering, Nagoya Institute of Technology. He received his B.S. degree in 1973, M.S. degree in 1975, and Ph.D. in 1981, all in Electrical and Electronics Engineering from the Faculty of Engineering, Nagoya University, Nagoya, Japan. He was an Alexander Humboldt Foundation Research Fellow at Giessen University, Federal Republic of Germany, in

1982-83. His current research interests are in the fields of neural network architecture, neurocomputing, application of neural networks, collaboration system development, and information security. Professor Iwata received the Best Paper Award of IEICE Trans. Inf. & Syst., Vol.J75-D-II, No.3, pp.545-553, 1992. He also received the Best Author Award of IPSJ, 1998. He is a member of the Japanese Neural Network Society, Japan Society of Medical Electronics and Biological Engineering, and the Information Processing Society of Japan. He is also a senior member of the IEEE.