16th Korea-Japan Joint Workshop on Frontiers of Computer Vision

Finding a sub-optimal combination of the binary classifiers for multi-class classification problems

Yoshito MURAYAMA†, Tetsu MATSUKAWA† and Takio KURITA‡

†: Graduate School of Systems and Information Engineering, University of Tsukuba
‡: National Institute of Advanced Industrial Science and Technology
{yoshito-murayam, t.matsukawa, takio-kurita}@aist.go.jp

Abstract: To use binary classifiers such as Support Vector Machines (SVM) for multi-class classification problems, the multi-class classifier is designed as a combination of binary classifiers in "one-versus-the-rest" or "one-versus-one" approach. In "Error-Correcting-Output-Code" approach, the classifier is designed such as the errors of some binary classifiers are collected by using the idea of the error correcting coding. However, there are so many other binary classifiers which can be used for multi-class classifier design and are not used in these methods. In this paper, we consider the problem to find the optimal combination of the binary classifiers. Since the number of subsets of the available binary classifiers is enormous, we use optimization technique such as Genetic Algorithm to find a sub-optimal combination. The experimental results show that the proposed approach achieves higher classification performances than the previous methods.

1 Introduction

In recent years, Support Vector Machine (SVM)[1] is applied for various classification problems. It is recognized that the SVM can produce high classification performance with high generalization. The SVM is an algorithm to construct a binary classifier. By introducing maximum margin technique to improve generalization performance, we can construct a binary classifier which has high classification accuracy for unknown samples. In addition, kernel techniques and soft margin technique make it possible to apply the SVM on nonlinear classification problems. To solve multi-class classification problems, however, we have to combine binary classifiers constructed by the SVM because the SVM can not directly construct multi-class classifier.

There are some typical approaches to design a multi-class classifier as a combination of binary classifiers such as "one-versus-the-rest" or "one-versus-one" approach[5]. In "Error-Correcting-Output-Code" (ECOC) [2], a multi-class classifier is designed so that the errors of some binary classifiers are corrected by using the idea of the error correcting coding. However, there are so many other binary classifiers which can be used for the multi-class classifier design and are not used in these methods. In this paper, we consider the problem to find the optimal combination of the binary classifiers. Since the number of subsets of the binary classifiers is enormous, we use the optimization technique such as Genetic Algorithm to find a sub-optimal combination.

2 The previous methods

2.1. "One-versus-the-rest"

In the "one-versus-the-rest" approach, a multi-class classifier is designed using binary classifiers that classify a certain class or the rest. The approach uses k classifiers for a k class classification problem. Fig. 1 shows an example of a multi-class classifier constructed by the "one-versus-the-rest". The advantage of this approach is that the number of necessary binary classifiers is the minimum to design a multi-class classifier among the previous methods. The "one-versus-the-rest" classifier can classify test samples, if only one of the k binary classifiers outputs $+1$. But in practice, there are two uncertain areas. One is the area that two or more classifiers outputs $+1$ (Uncertain Area 1 of Fig. 1) and the other is the area that all classifiers outputs -1 (Uncertain Area 2 of Fig. 1).

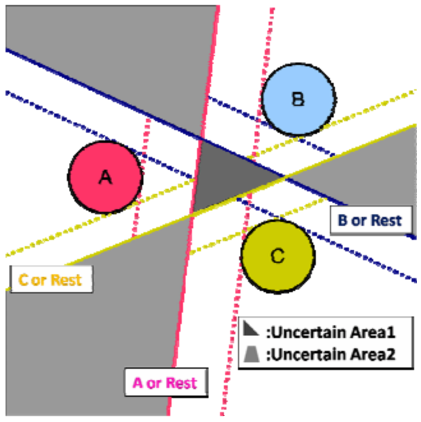


Figure 1: An example of a multi-class classifier constructed by the "one-versus-the-rest".

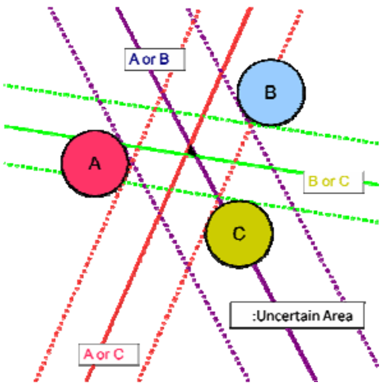


Figure 2: An example of a multi-class classifier constructed by the "one-versus-one".

To handle these uncertainties, we can classify a given sample to the class that is the maximum distance from the hyper planes. However, the validity of comparing the distances from hyper planes is not guaranteed because each classifier is independently constructed to solve an independent binary classification problem.

2.2. "One-versus-one"

The "one-versus-one" approach designs a multi-class classifier using all pair wise binary classifiers. This approach uses ${}_kC_2$ binary classifiers for a k class classification problem.

In the "one-versus-the-rest", there is only one classifier to classify each class, so there is no redundancy for each class. On the other hand, the "one-versus-one" uses multiple classifiers for each class, so this redundancy makes it possible to correct classification errors caused by a few classifiers. Fig. 2 shows an example of a multi-class classifier constructed by the "one-versus-one". In the "one-versus-one" approach, all binary classifiers vote on the class according to its classification results. Then the decision is made to the class that has the maximum number of votes. There is also an uncertainty problem in this approach when two or more classes have the same maximum number of votes (Uncertain Area of Fig. 2). In such case, the "one-versus-one" approach usually randomly selects the class or decides the class with the smaller class index.

2.3. "Error-correcting-output-code"

The "error-correcting-output-code" (ECOC) approach designs a multi-class classifier using the idea of the error correcting code. In the ECOC, we regard the k-class classification problems as the problems that transform the given unknown data $\mathbf{x}$ to the known class code C_k.

The errors of binary classifiers are considered as the errors of the transmission. The ECOC approach corrects the errors by the redundancy of the class codes. Table 1 shows an example of class codes for four classes.

Table 1: An example of Error-Correcting-Output-Code.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
C_1	1	1	1	1	1	1	1
C_2	0	0	0	0	1	1	1
C_3	0	0	1	1	0	0	1
C_4	0	1	0	1	0	1	0

The column represents a certain binary classifier. For example, the classifier f_1 outputs 1 for C_1 and 0 for the others. The classifier f_2 represents a classifier that classifies the two classes C_1, C_4 and the two classes C_2, C_3

The rows of Table 1 show the class code of a certain class. To correct the errors, the class codes are designed to separate from each other by the Hamming distance. The Hamming distance is defined as the number of different bits between codes. For the classification, the ECOC calculates the Hamming distance between output codes of the classifiers $f_1 \ldots f_7$ and the class codes. Then, the ECOC classifies to the class that is the nearest Hamming distance from the output code. For the example shown in the Table 1, if the binary classifiers of ECOC output the output code $\{1101010\}$, the Hamming distances between this output code and the classes codes of each class $\{C_1, C_2, C_3, C_4\}$ become $\{3,5,5,1\}$. Thus the ECOC classifies to the class C_4 because it is the nearest among 4 classes.

In the ECOC, the design of the class codes affects the classification performances of the multi-class classifier. Also the number of binary classifiers depends on the design of the class codes. There are some methods for the class code design. For example, there are "Exhausted Coding" [2], "BCH Code" [4] and so on. In this paper, we use "Exhausted Coding" for its simplicity.

Exhausted Coding designs the class codes that are $2^{k-1}-1$ bits. This means that the number of binary classifiers of this method is $2^{k-1}-1$. The class codes on Table 1 are designed by using Exhausted Coding. Representing the class codes as matrix like Table 1, Exhausted Coding designs the code of row i as alternating runs of 2^{k-i} zeroes and ones.

Although the number of binary classifiers used in the ECOC depends on the class code design, in the case of Exhausted Coding, it increases exponentially against the number of classes. This means that the number of binary classifiers is the largest in these three methods.

It is special that the ECOC uses n versus m classifiers that are not used in the "one-versus-the-rest" or the "one-versus-one" approach. Fig.3 shows an example of

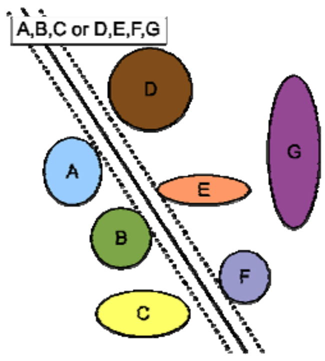


Figure 3: An example of n versus m classifier used in the ECOC.

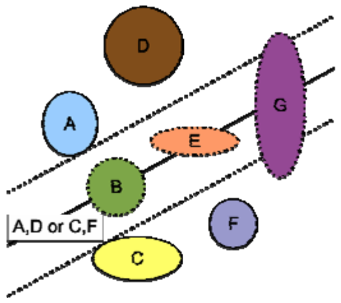


Figure 4: An example of n versus m classifier used in the proposed approach.

such binary classifier. The ECOC uses only n versus m classifiers that use all classes. But other binary classifiers that use a subset of all classes are not used in the ECOC.

3 Selection of binary classifiers for multi-class classification problems

3.1 Approach

There are many possible binary classifiers that can be used for multi-class classifier. In this paper, we propose a method that selects an optimal combination of binary classifiers from all considerable binary classifiers that can be used.

The n versus m classifiers are not used in the "one-versus-the-rest" and the "one-versus-one" approaches. In the ECOC, the one-versus-one classifiers are not used. Also there are the binary classifiers that are not used in these methods. Fig. 4 shows an example of such binary classifier. This is a n versus m classifier that is made from a subset of all classes, where n and m are more than two. By finding the optimal combination from all available binary classifiers, it is possible to design multi-class classifier that can achieve higher classification performance than the previous methods.

However, the number of such combinations increases as the number of the classes increases. Thus, it is hard to evaluate all possible combinations. In this paper, we propose an approach that searches a sub-optimal combination of binary classifiers from all possible classifiers by using Genetic Algorithm (GA). For a comparison, we also examine another approach that searches a sub-optimal combination from the binary classifiers that are used in the ECOC in the experiment in section 4..

To search a sub-optimal combination by GA, we need to represent multi-class classifiers as genes. In this paper, the gene represents whether certain binary classifier is used or not as a binary code $(1,0)$. When the number of binary classifiers that can be used is L, the gene is L bit length and represents whether each classifiers is used or not. Genetic operation is constructed from crossover and mutation. After genetic operation, the multi-class classifier design is changed.

3.2 Binary classifier

We use Support Vector Machine (SVM) in the following experiment as binary classifiers. However, our approach does not depend on what binary classifier is used

Let assume that we have N training samples as $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \mid y \in \{-1,+1\}\}$. Here, y is the class label that represents whether $\mathbf{x}$ belongs positive $(+1)$ or negative (-1).

To construct classifier that has a high generalization performance by SVM, we must properly select the hyper parameters such as the regularization parameter. Usually the hyper parameters are determined by evaluating the cross-validation.

3.2.1 Cross-validation

Cross-validation is defined as the following process.

[k fold cross-validation]

(1) Divide the training samples into k groups.

(2) Perform the following process recursively for $i = 1, \ldots, k$.

  (a) Learn classifier using the training samples except for i th group.

  (b) Test the classifier using i th group and get error rate r_i.

(3) Estimate test error rate as $\sum_{i=1}^{k} \frac{r_i}{k}$.

Note that the error rate r_i is given by $r_i = \frac{k * N_{error}}{(k-1) * N}$, where N is the number of samples and N_{error} is the number of misclassified samples. In this paper, the grid search is used to find the best hyper parameters.

Table 2: An example of class code that include "don't care" classes.

	f_1	f_2	f_3	f_4	f_5	f_6
C_1	1	1	1	X	X	X
C_2	0	1	X	1	1	1
C_3	X	0	1	0	X	1
C_4	X	X	0	X	0	0

3.3 Integration of the binary classifiers

Since a multi-class classifier is designed using many binary classifiers, it is necessary to integrate the results of each binary classifier to decide the output of the multi-class classifier. In this paper, we design class code and classify to the class that has the nearest Hamming distance from the output code of multi-class classifier.

However, "don't care" classes appear when we use the binary classifiers which use a subset of classes as shown in Fig 4. Table 2 shows an example of class codes in which such "don't care" classes are include. In this paper, we represent the class code of the classes that is not used to construct binary classifier as "don't care" class code X , and represent class code by the three codes $\{1,0,X\}$. The "don't care" class code means that this class is neglected (do not care) in Hamming distance calculation. For example, when the output code of an unknown data is $\{1,0,0,1,1,0\}$, the Hamming distance from each class code $\{C_1,C_2,C_3,C_4\}$ is calculated as $\{2,3,3,1\}$. Thus this unknown data is classified to C_4. If there are some classes that are the same Hamming distance, we classify it to the class that has the smallest class index.

3.4 Genetic algorithm

Genetic algorithm is one of the optimization techniques inspired by evolutionary biology. Genetic algorithm searches an optimal solution by three genetic operations such as the selection, the crossover and the mutation.

Let N_p be the number of parent population, N_c be the number of child population and G be the number of generations. The procedure of the Genetic algorithm is as follows;

[Genetic algorithm]

(1) Initialize N_p parents randomly.

(2) Select two individuals from the parents, and operate the crossover until the number of children reaches to N_c.

(3) Operate the mutation on children by mutation probability.

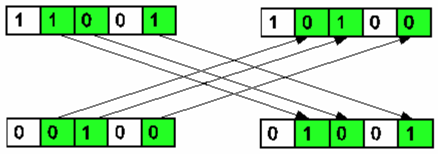


Figure 5: An example of crossover operation.

(4) Evaluate all the individuals.

(5) Select N_p individuals as new parents by the evaluation value.

(6) Repeat (2) ~ (5) until the number of iteration reaches to G.

One of the advantages of GA is its effectiveness for the problem that has a too large solution space.

3.4.1 Crossover

The crossover is an operation that generates children by replacing a part of the gene of the parents. Fig.5 shows that two children are generated by crossing over against the colored gene component. In this paper, the individuals in the parent set for crossover are selected at random. Crossover is defined as replacing gene components by crossover probability P_c for each gene component independently.

3.4.2 Mutation

To avoid falling into local optimal solutions, mutation changes components of the gene randomly. In this paper, we perform mutation at mutation probability P_m at each component of the gene independently.

3.4.3 Evaluation and selection

Evaluation is the operation that evaluates how does the individual fit as an optimal solution. Selection is the operation that selects the individuals as next parent. In this paper, we use classification rate $P_C = N_r / N_t$ as evaluation value for selection. Here, N_t is the number of training samples, N_r is the number of samples that is classified correctly. In the selection of individuals, $N_p/2$ individuals are selected by higher evaluation value, and the rest $N_p/2$ individuals are selected by roulette selection. Roulette selection selects individuals by individual selection probability

$$P_i = \frac{E_i}{\sum_{k=1}^{N_p/2+N_c} E_k}.$$

Where E_i is evaluation value of the i th individual.

Table 3: Details of datasets.

	Training samples	Test samples	Dimension of vector	Classes
Vehicle data	646	200	18	4
Satimage data	4435	2000	36	6

Table 4: Parameters of GA.

Parameters	Run time value
Number of parents N_p	500
Number of children N_c	500
Generations G	10000
Crossover probability P_c	0.5
Mutation probability P_m	0.01

4 Experiments

4.1. Experimental setup

We used Vehicle data and Satimage data from libSVM Datasets [3]. Table 3 shows the detail of the datasets. Linear SVM is used to train a binary classifier. For the implementation of SVM, we used libSVM [4]. Five-fold cross validation was carried out on the training set to tune the hyper parameters of SVM. In evaluation on GA, we used the training samples used to learn the SVM. Table 4 shows the parameters of GA used in experiments.

We evaluated two proposed approaches. One is the approach that searches a sup-optimal combination from the binary classifiers that are used in ECOC (approach A), the other is the approach that searches a sub-optimal combination from all the possible binary classifiers (approach B).

4.2. Experimental results

Table 5 shows the experimental result. The row of Table 5 shows the classification rates of each approach. The proposed approach A is the result of searching a sub-optimal combination from the binary classifiers that are used by ECOC. On the other hand, the proposed approach B is the result of searching a sub-optimal combination from all the possible binary classifiers. In Satimage data, the proposed approach A shows higher performance than the ECOC. However, the proposed approach A shows lower performance than the ECOC in the case of Vehicle data. This is because that the ECOC uses a few classifiers in Vehicle data that has only 4-classes. It seems there is no redundancy in the classifiers combinations in the ECOC for Vehicle data. On the other hand, the proposed approach B gives the highest performance for the both data because this approach can search the better classifiers from the larger combinations.

Table 5: Classification rates of each approach.

	1 vs 1	1 vs rest	ECOC	A	B
Vehicle data	0.795	0.77	0.785	0.765	0.84
Satimage data	0.851	0.735	0.785	0.841	0.86

Table 6: the number of binary classifiers of each approach.

	1 vs 1	1 vs rest	ECOC	A	B
Vehicle data	6	4	7	6	15
Satimage data	15	6	31	12	117

Table 6 shows the number of classifiers that each approach uses. The number of all possible binary classifiers is 26 for Vehicle data, and 302 for Satimage data. The proposed approach B uses the largest number of classifiers. For Satimage data, the proposed approach A could achieve higher classification performance than the ECOC with less number of binary classifiers.

Fig. 6 shows the distribution of classification results. In Fig.6, test samples are projected by principal component analysis (PCA) on the test samples. In Fig. 6, the white dots show misclassified samples and other color dots show correctly classified samples. The gray area shows the area where there are no samples. From Fig. 6 (a) ~ (d), it is noticed that there are many miss classification over class-2 on Satimage data. Table 7 shows classification rates of class-2 of Satimage data. For Satimage data, the “one-versus-the-rest” approach and the ECOC misclassify almost all samples of class-2. This is because the overlap area between class-2 and class-5,6 is too large to construct binary classifier that separates class-2 and class-5,6. In contrast, the approach like the “one-versus-one” that does not use such binary classifiers is effective in Satimage data. Since the proposed approach can automatically avoid to select such classifiers, the proposed approach is also effective in this data.

The uncertain area is explained in section 2. Here, we estimated the size of the uncertain area by counting uncertain samples that have the same Hamming distance from two or more classes. Table 8 shows the number of the uncertain samples. In the both of data, the proposed approach B shows the minimum number of uncertain samples. This is because that the proposed approach B can automatically select a better subset of binary classifiers depending on the properties of the training samples. Although ECOC uses more binary classifiers than “one-versus-one” approach, ECOC gives more number of uncertain samples than “one-versus-one”. One of the reasons seems that pair wise binary classifiers tend to have smaller uncertain area.

Table 7:Classification rates of class-2 on Satimage data.

	1 vs 1	1 vs rest	ECOC	A	B
Satimage data	0.34	0.02	0.03	0.40	0.44

Table 8:The number of uncertain samples of each approach.

	1 vs 1	1 vs rest	ECOC	A	B
Vehicle data	42	9	33	22	2
Satimage data	453	20	112	190	8

5 Conclusion

In this paper, we proposed an approach that designs multi-class classifier by selecting an optimal combination from the all possible binary classifiers. In the previous approaches such as the “one-versus-the-rest”, the “one-versus-all”, or the ECOC, the pre-designed combination of binary classifiers is used. In contrast, our approach searches a sub-optimal combination by adapting to the distribution of training data. Since the number of such combinations increases as to the number of the classes increases, we search a sub-optimal combination by genetic algorithm. From the result of experiment, it is confirmed that our approach can automatically design a multi-class classifier that can achieve higher classification performance than the previous approaches.

However, an enormous amount of computation time is necessary to construct all the possible binary classifiers that can be used for a multi-class classifier.

Also the computation time increases as to the number of classifiers increases. Thus, we want to deal with the problems that has large number of class in future work.

References

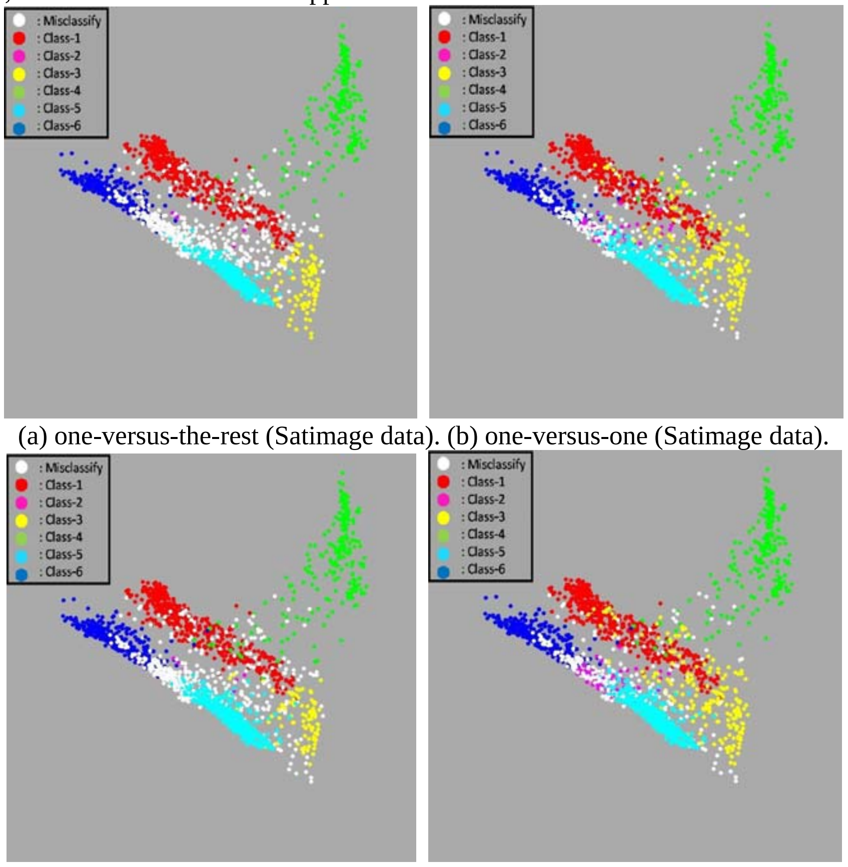
[1] V. N. Vapnik, Statistical Learning Theory, John Wiley & Sons(1998)

[2] Thomas G. Dietterich, Ghulum Bakiri, Solving Multi-class learning Problems via Error-Correcting Output Codes, Journal of Artificial Intelligence Research 2(1995)

[3] libSVM datasets:http://www/csie.ntu.edu.tw/ cjlin/libsvmtools/datasets

[4] Chin-Chung Chang, Chin-Jen Lin,{LIBSVM}a library for support vector machines, 2001, Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm

[5] Chih-Wei Hsu, Ching-Jen Lin: A comparison of Methods for Multi-class Support Vector Machine, IEE transaction on Neural Networks, Vol. 13, No. 2, 2002, pp. 415-425.



(a) one-versus-the-rest (Satimage data). (b) one-versus-one (Satimage data).

(c) ECOC (Satimage data). (d) proposed approach B (Satimage data).

Figure 6: Distribution of result of classification over Satimage dataset on test data.
For better viewing, please see the color pdf file.