Decision Tree Using Class-Dependent Feature Subsets

Kazuaki Aoki and Mineichi Kudo

Division of Systems and Information Engineering, Graduate School of Engineering
Hokkaido University, Kita-13, Nishi-8, Kita-ku, Sapporo 060-8628, Japan
{kazu,mine}@main.eng.hokudai.ac.jp
http://ips9.main.eng.hokudai.ac.jp

Abstract. In pattern recognition, feature selection is an important technique for reducing the measurement cost of features or for improving the performance of classifiers, or both. Removal of features with no discriminative information is effective for improving the precision of estimated parameters of parametric classifiers. Many feature selection algorithms choose a feature subset that is useful for all classes in common. However, the best feature subset for separating one group of classes from another may depend on groups. In this study, we investigate the effectiveness of choosing feature subsets depending on groups of classes (class-dependent features), and propose a classifier system that is built as a decision tree in which nodes have class-dependent feature subsets.

1 Introduction

Feature selection is to find a feature subset that is effective for classification from a given feature set. This technique is effective for both improving the performance of classifiers and reducing the measurement cost of features. Particularly when the scale of the problem is large (in the sense of the number of features or the number of classes, or both), there are some features that have little or no discriminative information. It is well known that such features, called *garbage features*, weaken the performance of classifiers (peaking phenomenon) as long as a finite number of training samples is used for designing the classifiers. Thus, removal of such garbage features should result in an improvement in the performance of such classifiers.

Many techniques for feature selection have been proposed [1,2,3,4,5,6,7]. All of these approaches choose the same feature subset in all classes. However, it seems reasonable to assume that effective feature subsets are different, depending on classes. For instance, in the case of more than two classes, a feature subset that is powerful in discriminating one class from the remaining classes does not always work in discriminating another class from the remaining classes. Thus, when treating many classes, such as in character recognition, selection of feature subsets depending on groups of the classes is effective. We call such a feature subset a *"class-dependent feature subset."*

T. Caelli et al. (Eds.): SSPR&SPR 2002, LNCS 2396, pp. 761–769, 2002.

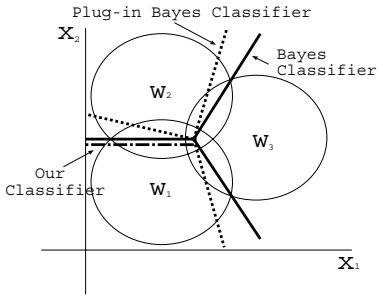

Fig. 1. Example

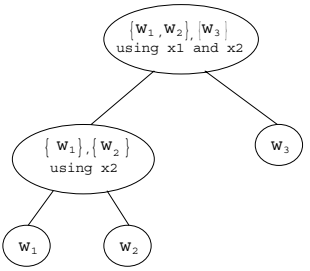

Fig. 2. Decision tree for Fig.1

For instance, in Chinese character recognition, there are more than one thousand characters. A group of similar characters has almost the same values in almost all features but differs in a small number of features, e.g., the number of strokes or whether a short stroke exists or not. Only a small number of features are effective in discrimination of these similar characters. However, these features are not always useful for discrimination between the group of these similar characters and other groups of characters.

Therefore, it is expected that the performance of classifiers can be improved by choosing feature subsets depending on groups of classes. In fact, there have been some studies in which class-dependent features worked well in handwritten character recognition [8,9]. However, theoretical analysis is expected. In this paper, we present a formalization of the usage of *class-dependent feature subsets* and propose a classification system using these subsets.

2 Illustrative Example

Our concept is explained by the example shown in Fig. 1. In Fig. 1, there are three classes according to normal distributions with the same covariance matrix and different means. The Bayes rule therefore becomes linear. As long as a given training sample is finite, we cannot avoid misclassification in the plug-in Bayes classifier estimated from the training sample. However, we can reduce such a misclassification using class-dependent feature subsets. Indeed, in this problem, only feature x_2 has discriminative information between ω_1 and ω_2. Thus, a classifier using only x_2 is expected to perform better than that using x_1 and x_2 in this case. A decision tree designed naturally is shown in Fig. 2.

3 Decision Trees

3.1 Several Types of Decision Tree

We described how class-dependent feature subsets are used to improve classifiers. Then, the next question is what uses of class-dependent feature subsets

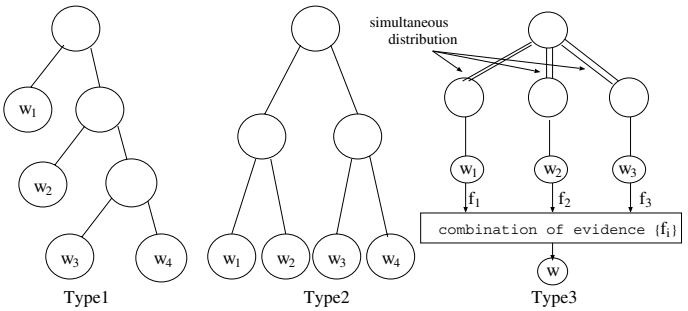

Fig. 3. Three types of decision tree

are possible. Some decision trees are naturally considered (Fig. 3). In all these configurations, the recognition process starts from a whole set of classes at a root down to subsets of classes and reaches a single class at a leaf. Each type has the following aspects.

(*Type* 1) This type of decision tree has the simplest architecture and separates one class from the other classes in each node. The problem shown in Fig. 1 can be solved using this type of decision tree.

(*Type* 2) This is a generalization of type 1. In each node, data are separated into two subsets of classes.

(*Type* 3) The process goes from the root to its children in parallel, and in each child one class is separated from the other classes. Usually, each node outputs a value of the evidence about how well the decision is firm. In the process of gathering evidence, the final decision is made.

Type 3 turns into regular classifiers like a linear or quadratic classifier in multi-class cases when all nodes have the same feature subsets and the evidence is combined by the maximum likelihood method. Another approach called a modular network [8,9] is also included in this type.

In each node of a decision tree, the problem is to classify one group of classes and another group of classes. Here, let two groups of classes be Ω^1 and Ω^2. Then, a node is identified by the following information.

1. (Ω^1, Ω^2): two groups of classes to be classified
2. F: feature subset
3. ϕ: classifier

Thus, an internal node t is denoted as

$$Node^t = \{\Omega_t^1, \Omega_t^2, F_t, \phi_t\}.$$

Our approach is different from conventional decision tree approaches [10,11] in the following two points:

1. In conventional decision tree approaches, each node is split in terms of *impurity* in a feature, whereas in our approach, each node is split in terms of the degree of separation between two groups of classes.
2. In conventional decision tree approaches, classification in each node is simple, and usually only one feature is used. In other words, the performance of classification in each node is not so good. Instead, those approaches complement the simplicity with splitting of data many times. In our approach, we split each node by a class-dependent feature subset. Thus, it is expected that the classification performance is improved in individual small problems.

In this paper, we considered only type 1 and type 2 decision trees.

3.2 Experiments Using Artificial Data 1

We examined the possibility of this approach under the assumption that a decision tree is ideally constructed and ideal class-dependent feature subsets are chosen. The data are shown in Table 1. In Table 1, a feature with 1 means that data are generated according to a normal distribution with average 0.5 and standard deviation 0.1 in the feature, and a feature with 0 means that data are generated according to a normal distribution with average 0 and standard deviation 0.1. There is no correlation between features. For example, it is sufficient to use only features x_1 and x_2 to separate ω_1 and ω_2. As a whole, all features are needed to classify this dataset.

In the ideal tree, at the root node ω_1 is separated from the others by all features $x_1 - x_c$, and then ω_2 by $x_2 - x_c$, and so on (Fig. 4), where c is the number of classes and is also the number of features. In every nodes, a linear classifier was used. Compared with the linear classifier with the full feature set, this tree can estimate the parameters accurately in the deeper nodes.

The results of the experiment are shown in Fig. 5. In the experiment, three, ten or thirty classes and 5, 10, 100 or 1000 training samples per class were used. The number of test samples was fixed at 1000 per class. The average recognition rate of 10 different training datasets is shown. For comparison, the recognition rate by the linear classifier with all features is shown. It was found that the decision tree with class-dependent feature subsets worked better than the linear classifier with all features, when the number of training samples was comparatively small and the number of classes, and also the number of features, was large.

4 Construction of Decision Trees

In this section, we propose an algorithm to construct a decision tree from given data. A decision tree is constructed in a bottom-up way like Huffman coding. The algorithm is as follows.

1. Initialization step: Set $\Omega_i = \{\omega_i\}, (i = 1, 2, \cdots, C)$, $c = C$,$t = 1$. Attach an unprocessed mark to all Ω_i. These Ω_i correspond to leaves.

Table 1. Artificial data

	features				
class	x_1	x_2	x_3	$\cdots$	x_c
ω_1	1	0	0	$\cdots$	0
ω_2	0	1	0	$\cdots$	0
ω_c	0	0	0	$\cdots$	1

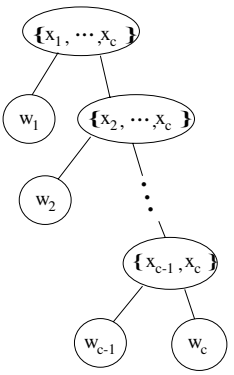

Fig. 4. Decision tree used in this experiments

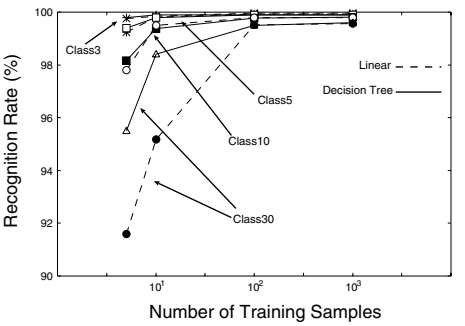

Fig. 5. Result of experiments using data shown in Table.1

2. Calculate the separability S_{ij} of pair (Ω_i, Ω_j) for all unprocessed nodes Ω_i and Ω_j, $(i, j = 1, \cdots, c)$.
3. Choose the pair $(\Omega_{i^*}, \Omega_{j^*})$ with the smallest separability $S_{i^*j^*}$. Let Ω_{i^*} be Ω^1_{c+1} and Ω_{j^*} be Ω^2_{c+1}. Mark Ω_{i^*} and Ω_{j^*} as processed. Select a feature subset F_{c+1} that is effective in discrimination between Ω^1_{c+1} and Ω^2_{c+1}.
4. Construct a classifier ϕ_{c+1} to classify Ω^1_{c+1} and Ω^2_{c+1} with feature subset F_{c+1}. In this step, we have a new node, $Node^{c+1} = \{\Omega^1_{c+1}, \Omega^2_{c+1}, F_{c+1}, \phi_{c+1}\}$.
5. $\Omega_{c+1} = \Omega^1_{c+1} \bigcup \Omega^1_{c+2}$ and $c \leftarrow c + 1$üA$t \leftarrow t + 2$.
6. Repeat steps 2-5 until $t = c$.

In steps 3-5, two nodes are merged into one new node, $(c \leftarrow c + 1)$, and the two merged nodes are marked as being processed, $(t \leftarrow t + 2)$. Finally, a decision tree with $2C - 1$ nodes is constructed.

5 Experiments

We dealt with 4-class and 9-class artificial datasets shown in Fig.6 and real 'mfeat' data from UCI Machine Learning Database [12].

5.1 Artificial Data 2

The separability measure, the classifier, and the feature selecton method we used are as follows:

1. Separability: the recognition rate estimated by the leave-one-out technique with 1-NN (the nearest neighbor) classifier.
2. Feature selection method: an approach based on the structual indices of categories [13] .
3. Classifier: plug-in Bayes linear classifiers

In this experiment, the same type of classifiers were used in all nodes but trained differently in each node.

In this dataset, we compared two decision trees: (1) a decision tree with ideal feature subsets and with an ideal configuration and (2) a decision tree constructed by the proposed algorithm. We dealt with the same type of problems in 4-class and 9-class cases (Fig. 6). The number of training samples was 10 per class, and the number of test samples was 1000 per class. The constructed decision tree is shown in Fig. 7, and the classification boundaries are shown in Fig. 8 and Fig. 9. The figures in parentheses in those figures show the recognition rates for test samples.

The ideal boundary and the boundary by the proposed method are almost comparable. Here, it should be noted that if a single feature is used, the boundary becomes a straight line regardless of the classifier used. We succeeded in improving the performance of the decision tree with the differently trained linear classifiers, compared to that of the single linear classifier using all features.

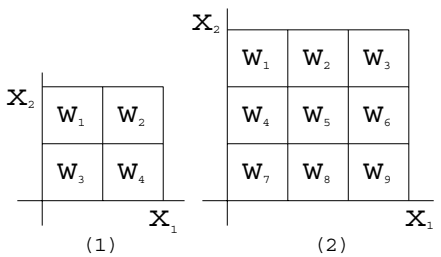

Fig. 6. (1) 4-class and (2) 9-class problem

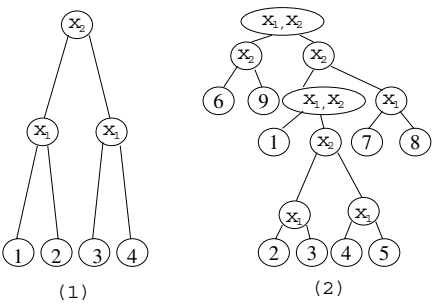

Fig. 7. Constructed decision trees (1) in a 4-class problem and (2) in a 9-class problem

5.2 'mfeat' Data

Next, we examined the mfeat data. This dataset is a handwritten numeric '0'-'9' database. The feature dimension is 76, and the number of training samples is

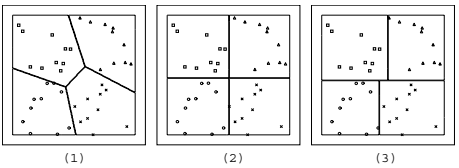

Fig. 8. Constructed boundary in a 4-class problem: (1) Linear (90.4%), (2) Ideal decision tree (97.5%), and (3) The proposed method (92.8%)

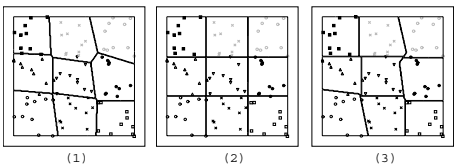

Fig. 9. Constructed boundary in a 9-class problem: (1) Linear (91.2%), (2) Ideal decision tree (97.8%), and (3) The proposed method (93.5%)

200 per class. Those features are Fourier coefficients of a character shape. The top 100 samples of each class were used for training, and the bottom 100 samples were used for testing. The same way as Artificial DAta 2 for the decision tree. The constructed decision tree is shown in Fig. 10. For comparison, we constructed a linear classifier and a 1-NN classifier with all features. The recognition rate of the linear classifier was 80.7%, that of the 1-NN classifier was 82.4%, and that of the proposed method was 83.8%. The 1-NN classifier attained 83.3% when a feature subset common to all classes was chosen.

From the decision tree, we can see that the number of selected features is appropriate depending on the problem. For instance, in node classifying class '0' and '8', the local problem is easy to solve and the number of features is very small. The effectiveness of our approach depends on the number of training samples and the number of classes. Thus, this approach, like other approaches, does not always work well for all kinds of datasets. It is expected that this approach will work well for problems in which the number of classes is large.

6 Discussion

Our approach using class-dependent feature subsets works effectively when the number of classes is large, and it is superior to conventional approaches using a common feature subset. It is expected that it works better when the number of training samples is smaller, because less features take advantage in such a case and individual small problems with only a few classes require less features than the total problem with many classes.

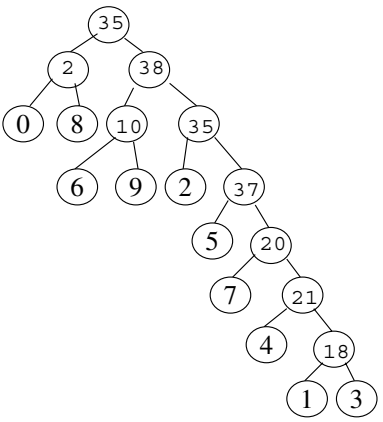

Fig. 10. Decision tree for mfeat (figures in internal nodes are the numbers of features)

Another merit of our approach is that the results are interpretable. In each node, we can show how easy it is to solve a local problem and what set of features is necessary. With such information, it should be possible to improve the performance of the classifier.

7 Conclusion

We have discussed the effectiveness of a class-dependent feature subset, and have presented an algorithm of a classification system as a decision tree. In addition, we can see separability in each node of the decision tree, so we may use this information to improve the decision tree. We will consider the design of the optimum decision tree.

References

1. P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Prentice-Hall, 1982. 761
2. P. Pudil, J. Novovičová and J. Kittler, Floating Search Methods in Feature Selection. *Pattern Recognition Letters*, **15**(1998), 1119–1125. 761
3. P. Somol, P. Pudil, J. Novovičová and P. Paclík, Adaptive Floating Search Methods in Feature Selection. *Pattern Recognition Letters*, **20**(1999), 1157–1163. 761
4. F. J. Ferri, P. Pudil, M. Hatef and J. Kittler, Comparative Study of Techniques for Large-Scale Feature Selection. *Pattern Recognition in Practice IV*(1994), 403–413. 761
5. M. Kudo and J. Sklansky, A Comparative Evaluation of Medium- and Large-scale Feature Selectors for Pattern Classifiers. *1st International Workshop on Statistical Techniques in Pattern Recognition*(1997), 91–96. 761

6. M. Kudo and J. Sklansky, Classifier-Independent Feature Selection for Two-stage Feature Selection. *Advances in Pattern Recognition*, **1451**(1998), 548–554. 761
7. D. Zongker and A. Jain, Algorithms for Feature Selection: An Evaluation. *13th International Conference on Pattern Recognition*, **2**(1996), 18–22. 761
8. I. S. Oh, J. S. Lee and C. Y. Suen, Analysis of Class Separation and Combination of Class-Dependent Features for Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(1999), 1089–1094. 762, 763
9. I. S. Oh, J. S. Lee, K. C. Hong and S. M. Choi, Class Expert Approach to Handwritten Numerical Recognition. *Proceedings of IWFHR '96*(1996), 35–40. 762, 763
10. R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification: Second Edition*. John Wiley & Sons, 2000. 763
11. L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*. Wadsworth & Brooks / Cole Advanced Books & Software, 1984. 763
12. P. M. Murphy and D. W. Aha, UCI Repsitory of Machine Learning Databases [Machine-readable data repository]. *University of California Irnive, Department of Informaiton and Computations Science*(1996). 765
13. M. Kudo and M. Shimbo, Feature Selection Based on the Structural Indices of Categories. *Pattern Recognition*, **26**(1993), 891–901. 766