한국지능정보시스템학회논문지 제10권 제2호 2004년 11월(pp.133~145)

Hybrid Multiple Classifier Systems

In-cheol Kim
Department of Computer Science Kyonggi University
(kic@kyonggi.ac.kr)

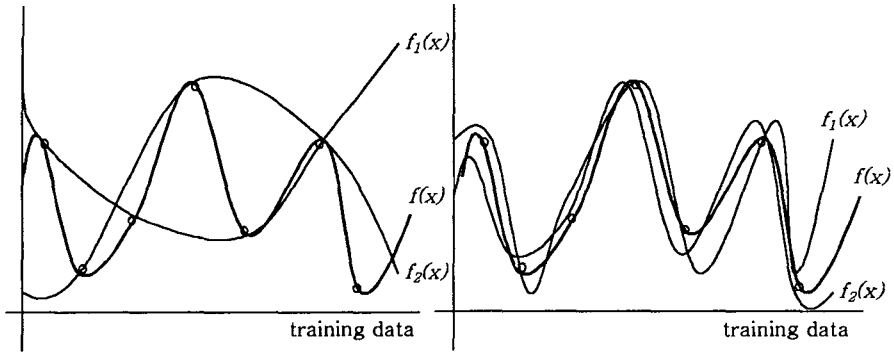
Combining multiple classifiers to obtain improved performance over the individual classifier has been a widely used technique. The task of constructing a multiple classifier system(MCS) contains two different issues : how to generate a diverse set of base-level classifiers and how to combine their predictions. In this paper, we review the characteristics of the existing multiple classifier systems: *bagging*, *boosting*, and *stacking*. And then we propose new MCSs: *stacked bagging*, *stacked boosting*, *bagged stacking*, and *boosted stacking*. These MCSs are a sort of hybrid MCSs that combine advantageous characteristics of the existing ones. In order to evaluate the performance of the proposed schemes, we conducted experiments with nine different real-world datasets from UCI KDD archive. The result of experiments showed the superiority of our hybrid MCSs, especially *bagged stacking* and *boosted stacking*, over the existing ones.

Key words : Multiple Classifier System, Bagging, Boosting, Stacking, Meta Learner, Bias, Bootstrap Sampling

Received: August 2004 Accepted: November 2004 Corresponding Author: In-cheol Kim

1. Introduction

Recently, several multiple classifier systems (MCSs) have been explored in an effort to obtain improved performance by combining weak classifiers(Ali et al, 1996). In many practical applications, such as pattern recognition and text data mining, the MCS approach was reported to be effective. In order to build a MCS, we can derive multiple different classifiers from different versions of the training data set or different types of learning algorithms. These derived classifiers can be then combined in various ways such as voting or meta learning. In this paper, we briefly review the characteristics of the existing MCSs: *bagging*(Bauer et al, 1999), *boosting*(Dietterich, 2000), and *stacking*(Breiman, 1996). And then we propose new MCSs such as *stacked bagging*, *stacked boosting*, *bagged stacking*, and *boosted stacking*. These MCSs are a sort of hybrid MCSs that combine advantageous characteristics of the existing ones. Through several experiments with nine different real-world datasets from UCI KDD archive(Blake et al, 1998), we compare the performance of the hybrid MCSs with that of the existing ones.



(a) Locally approximate functions f_1, f_2 (b) Globally approximate functions f_1, f_2

[Fig. 1] Varying training dataset vs using different learning algorithms

2. Multiple Classifier Systems

Multiple classifier systems(MCSs), which are also called *classifier ensembles,* refer to a collection of methods that learn a target function by training a number of individual classifiers and combining their predictions. Multiple classifier systems combine a set of redundant classifiers to improve reliability and accuracy. Design of a MCS has two basic issues: how do we create the individual classifiers?; how do we perform the combination of these classifiers?

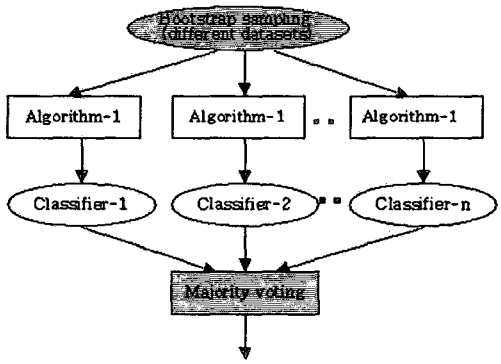
There are several ways of creating different classifiers. The first is to vary the training dataset and then apply a single learning algorithm to these different versions of training dataset. This is the mostly used approach to produce different classifiers. We can vary the training data by using different sampling methods such as random sampling without replacement or random sampling with replacement - also called *bootstrap sampling.* This approach makes use of different variances of the training datasets. Another way of creating different classifiers is to vary the learning algorithm. Each learning algorithm has a unique bias on knowledge representation and search procedure. Therefore, applying different learning algorithms to the same training data result in different classifiers. For example, while the C4.5 algorithm generates a decision tree as classifier, the BPN algorithm learns a neural network as classifier. This approach can be viewed to make use of the different biases of the learning algorithms. The last possible way to create different classifiers is to vary only learning parameters such as the initial weights in an MLP without any changes of the training data and the learning algorithm themselves. [Fig. 1] compares (a) varying the training dataset with (b) using different learning algorithms. The different classifiers generated by varying the training dataset correspond to locally approximate functions f_1 and f_2 of the target function f shown in [Fig. 1] (a). The classifiers derived by applying

different learning algorithms, however, correspond to globally approximate functions f_1 and f_2 shown in [Fig. 1] (b).

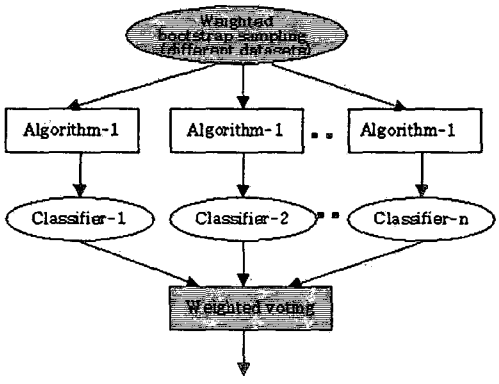
There are also several ways of combining the different classifiers. The first is so-called *best classifier* method, in which one of the classifiers is selected as the best and then its prediction is output as the final prediction of the system. The second is *majority voting*, which combines the predictions of all classifiers equally. The third is *weighted voting*, which combines the predictions of classifiers depending on their weights. The last is using a meta learner, which learns the final decision from the predictions of the base classifiers.

The current well-known multiple classifier systems are *bagging*(Bauer et al, 1999), *boosting* (Dietterich, 2000), and *stacking*(Breiman, 1996). Basically both bagging and boosting create different classifiers by varying the training dataset. They commonly use *bootstrap sampling*. Bootstrap sampling can be formulated as the following. Given a training data set $L = \{(x_n, y_n), n = 1,..., N'\}$, where x_n is the nth instance represented as a vector of its attribute values and y_n is its class value, *bootstrap sampling* randomly samples L with replacement into K subsets of size N, where $N \leq N'$.

[Fig. 2] and [Fig. 3] illustrate the concept of bagging and boosting, respectively. Bagging uses bootstrap sampling in order to obtain different versions of a given data set. The size of each sampled data set equals the size of the original data set. On each of these versions of the data set, the same learning algorithm is applied. Classifiers obtained in this manner are then combined with majority voting. On the other hand, boosting uses weighted bootstrap sampling to obtain different versions of a given data set and then applies weighted voting to combine the derived classifiers. Boosting first builds a classifier with some learning algorithm from the original data set. Then the weights of the examples misclassified by the classifier are increased. Based on the changed



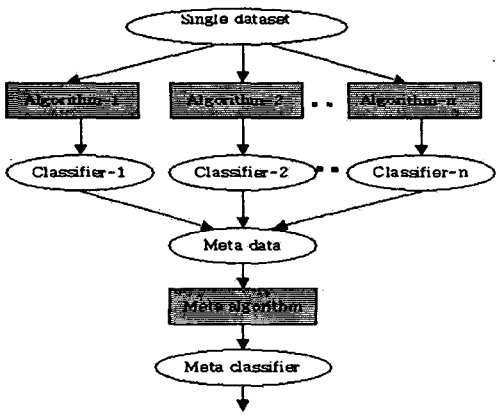
[Fig. 2] Bagging



[Fig. 3] Boosting

weights, a new version of the training dataset is derived through weighted sampling. And then by applying the same learning algorithm to the new version of the training dataset, another classifier is generated. The procedure is repeated several times. Classifiers generated in this manner are then combined using weighted voting, in which each classifier is weighted in proportion to its classification accuracy. In the phase of classifier generation, both bagging and boosting use different versions of the training dataset and in the phase of classifier combination, they use voting.

[Fig. 4] illustrates the concept of stacking, which is another well-known form of MCS. In stacking, different learning algorithms are applied to the same data set in order to obtain different base-level classifiers. Then a meta-level learning algorithm is applied to learn the final classification result from the predictions of the base-level classifiers. The meta-classifier generation process is like the following. We use the training data set $L = \{(x_n, y_n), n = 1,..., N'\}$ as a test set for each of the K *base-level* or *level-0 classifiers*. Suppose that there are I output classes, and let $P_{ki}(x)$ denote the probability that the kth classifier assigns to the ith class given the test instance x. The vector $P_{kn} = (p_{k1}(x_n),..., p_{ki}(x_n),..., p_{kI}(x_n))$ gives the kth classifier's class probabilities for the nth instance, and at the end of the testing process, the data assembled from the output of the K classifiers is $L' = \{(P_{1n},..., P_{kn},..., P_{Kn}, y_n), n = 1,..., N'\}$. This is called *meta data* or *level-1 data*. Use a learning algorithm, which we call the *meta learner* or *level-1 generalizer*, to derive a classifier M' that predicts the class from this meta data. M' is called the *meta classifier* or *level-1 classifier*. To classify a new instance, the base-level classifiers M_k are used to produce a vector $(p_{11},.., p_{1I},.., p_{k1},.., p_{kI},.., p_{K1},.., p_{KI})$ which is input to the meta classifier M', and the output of M' is the final classification result for that instance. Different from other MCSs such as bagging and boosting, stacking uses different learning algorithms in the phase of classifier generation, and then it uses meta learning in the phase of classifier combination.

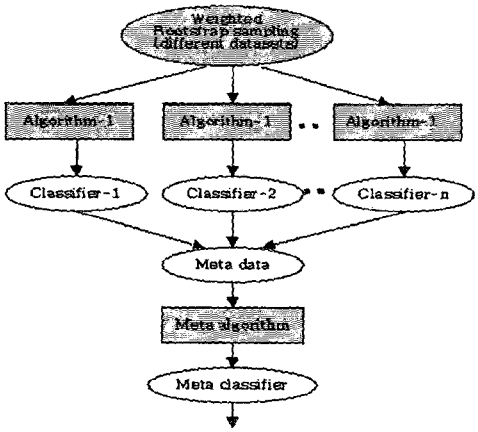


[Fig. 4] Stacking

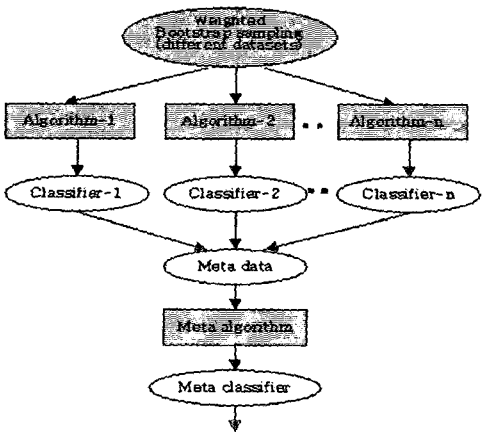
3. Hybrid Multiple Classifier Systems

In this study, we develop two classes of hybrid schemes: the one is the extended bagging and the extended boosting which employ meta learning to combine the predictions of base-level

classifiers, and the other is the extended stacking which uses different versions of the data set to create base-level classifiers. As mentioned above, pure bagging and pure boosting create multiple homogenous base-level classifiers by varying the training data set, but combine the predictions of these classifiers by simple voting. *Stacked bagging* and *stacked boosting* correspond to the extended bagging and the extended boosting, respectively.



[Fig. 5] Stacked boosting



[Fig. 6] Boosted stacking

They replace voting with meta learning to combine predictions of the base-level classifiers. Generally speaking, meta learning has higher flexibility than simple voting. [Fig. 5] illustrates the concept of stacked boosting, which is one of the hybrid MCSs built in this manner. Stacked boosting uses the weighted bootstrap sampling to obtain different versions of the training data set. Then it applies a base-level learning algorithm to these different versions of the training data set in order to generate different base-level classifiers. After preparing meta data from predictions of the base-level classifiers, it applies a meta-level learning algorithm to this meta data in order to generate a meta classifier. To classify a new instance, the base-level classifiers are used to produce a vector of class probabilities of this instance which is input to the meta classifier, and the output of the meta classifier is the final classification result for that instance.

Pure stacking creates multiple heterogeneous base-level classifiers by applying different learning algorithms to the same data set, and then combines predictions of these base-level classifiers by meta learning. Although pure stacking takes advantages of different biases of the learning algorithms and flexibility of the meta learning, it does not make use of different variances of the training dataset. *Bagged stacking* and *boosted stacking* are the extended stacking schemes, which create multiple base-level classifiers by applying different learning algorithms to different versions of the training data set. In order to obtain different versions of

<Table 1> Boosted stacking algorithm

Base Classifier Generation :
Assign equal weight to each training instance
For each of k iterations
Apply the k-th base learning algorithm to weighted dataset
and store the resulting base classifier
Compute error e of model on weighted dataset and store error
If e equals to zero, or e greater or equal to 0.5
Terminate classifier generation
For each instance in dataset
If instance classified correctly by classifier
Multiply weight of instance by $e/(1-e)$
Normalize weight of all instances

Meta Data Generation :
For each instance in dataset
For each of k iterations
Apply the k-th base classifier to predict the class probability of the instance
Generate a meta data by combining class probabilities from the base classifiers

Meta Classifier Generation :
Apply the meta learning algorithm to the meta dataset
and store the resulting meta classifier

Classification :
For each of k iterations
Apply the k-th base classifier to predict the class probability of the new instance
Generate a test data for the meta classifier by combining k predictions of the base classifiers
Return the class predicted by the meta classifier

the data set, *bagged stacking* and *boosted stacking* use the simple bootstrap sampling and the weighted bootstrap sampling, respectively. [Fig. 6] illustrates the concept of boosted stacking and <Table 1> summarizes the boosted stacking algorithm. Note that in stacked bagging and stacked boosting the base-level classifiers are homogenous ones derived from the same learning algorithm, but in bagged stacking and boosted stacking the base-level classifiers are heterogeneous ones derived from different learning algorithms. This is the only difference between two classes of hybrid MCSs.

4. Experiments

In this section, we explain the experiments to compare the hybrid MCSs with the existing standard MCSs. For our experiments, nine different real-world datasets from UCI KDD archive were used. They are balance scale, breast cancer, glass, heart disease, hepatitis domain, ionosphere, iris, vote, soybean data sets. C4.5, k-Nearest Neighbors(k-NN), and Naïve Bayesian(NB) learning algorithms were applied to obtain single classifiers and base-level classifiers. In bagging and boosting, only C4.5 and NB

learning algorithms were used to build homogenous base-level classifiers. In stacking, all three different algorithms(C4.5, k-NN, and Naïve Bayesian(NB)) were applied to generate heterogeneous base-level classifiers, but only two of them, C4.5 and NB, were used as meta learner.

The experiments were performed on a computer system equipped with Linux operating system, Intel Pentium IV processor, and 512 MB of RAM memory. <Table 2-1> and <Table 2-2> summarize the result of our experiments.

<Table 2-1> Experimental results: classification accuracy (%)

Schemes	Data Sets								
	balance	breast	glass	heart	hepatitis	Ionosphere	iris	vote	soybean
C4.5	74.41	93.70	61.64	77.67	80.77	80.67	92.16	94.62	89.70
k-NN	76.30	95.80	64.38	80.58	80.77	86.55	96.08	93.24	87.98
NB	87.68	95.38	46.58	84.47	82.69	83.19	96.08	91.22	90.99
Bagging (B:C4.5)	80.09	96.22	69.86	80.58	80.77	87.39	96.08	96.30	92.13
Bagging (B:NB)	87.68	95.38	53.42	84.47	82.69	81.51	96.08	91.22	90.99
Boosting (B:C4.5)	81.30	94.96	65.75	82.52	82.69	90.76	96.08	96.62	91.42
Boosting (B:NB)	89.10	95.38	47.95	83.50	80.77	88.24	96.08	96.65	90.13
Stacking (M:C4.5)	88.63	95.80	58.90	87.38	80.77	85.71	96.08	95.95	92.27
Stacking (M:k-NN)	84.36	94.96	49.32	79.61	80.77	83.19	96.08	96.62	92.27
StackedBag (B:NB,M:NB)	88.96	95.99	65.01	80.54	82.58	81.77	96.00	89.66	90.04
StackedBag (B:C4.5,M:B)	84.64	95.85	63.08	80.53	70.71	91.17	94.00	95.86	93.12
StackedBag (B:k-NN,M:NB)	90.88	95.99	69.16	77.23	78.71	87.46	96.67	93.10	90.34
StackedBag (B:NB,M:C4.5)	90.24	95.56	41.59	82.18	86.45	88.89	93.33	90.11	88.58
StackedBag (B:C4.5,M:C4.5)	82.56	93.84	62.62	75.25	80.65	87.75	94.67	96.09	82.87
StackedBag (B:k-NN,M:C4.5)	86.24	95.42	66.35	76.90	80.00	88.03	95.33	93.56	90.78

(B:base learner, M:meta learner)

<Table 2-2> Experimental results: classification accuracy (%)

Schemes	Data Sets								
	balance	breast	glass	heart	hepatitis	Ionosphere	iris	vote	soybean
StackedBoost (B:NB,M:NB)	87.84	96.28	33.64	79.31	84.51	92.02	95.33	93.79	88.87
StackedBoost (B:C4.5,M:NB)	85.68	95.56	62.15	79.54	78.06	90.31	94.00	94.02	92.83
StackedBoost (B:k-NN,M:NB)	82.24	94.85	66.35	71.95	75.48	86.32	95.33	91.95	88.43
StackedBoost (B:NB,M:C4.5)	93.92	95.28	59.34	81.19	81.29	91.17	95.33	94.94	86.68
StackedBoost (B:C4.5,M:C4.5)	87.76	95.71	70.09	74.92	79.35	90.88	93.33	96.55	91.80
StackedBoost (B:k-NN,M:C4.5)	86.48	95.99	60.28	75.58	80.00	87.46	98.00	90.80	87.70
BaggedStack (M:NB)	86.24	95.85	61.01	82.62	81.94	91.45	96.00	96.78	92.53
BaggedStack (M:C4.5)	90.40	96.14	63.55	82.51	83.29	91.74	94.00	96.09	91.22
BoostedStack (M:NB)	92.32	96.14	63.03	83.17	81.29	90.31	95.33	95.03	93.99
BoostedStack (M:C4.5)	92.96	96.14	69.15	82.18	83.35	89.74	96.00	97.63	93.80

(B:base learner, M:meta learner)

The experimental results show clearly that any types of MCS outperform single classifiers over almost all datasets. <Table 3> that lists average accuracies of different MCSs also confirms our finding. Both pure bagging(Av_Bag) and pure boosting(Av_Boost) showed higher performance than single classifiers(Av_Single) over all datasets. On some datasets, however, pure stacking(Av_Stack) sometimes showed lower performance than single classifiers(Av_Single). One of the reasons may be that the applied learning algorithms generated low-performance classifiers on these datasets and the classifiers did not much differ from each other in terms of performance. In such cases, stacking schemes based on different biases of the learning algorithms and meta learning may be less effective. Through experimental results, we can also find that hybrid MCSs outperform the existing ones in many cases. When we compare average classification accuracies among pure stacking(Av_Stack), bagged stacking(Av_BagStack) and boosted stacking(Av_BoostStack), we can find that both bagged stacking and boosted stacking showed higher performance than pure stacking. However, stacked bagging and stacked boosting did not show the improved performance compared with pure bagging and pure boosting, respectively. This

implies that in hybrid MCSs, it is clearly effective to make use of different versions of training dataset. But it is not clear that meta learning for combining base-level classifiers is effective.

In order to analyze the effectiveness of the sampling methods, we compare average accuracies among pure bagging(Av_Bag), pure boosting (Av_Boost), bagged stacking(Av_BagStack), and boosted stacking(Av_BoostStack). Pure boosting and boosted stacking are slightly superior to pure bagging and bagged stacking, respectively. This implies that the weighted bootstrap sampling is more effective than the simple bootstrap sampling to make a variety of base-level classifiers. We also try to find out which one of two learning algorithms(C4.5 and NB) is better as meta learner, but our experimental results do not give any distinct answer.

<Table 4> summarizes classification accuracies of single classifiers and stacking schemes. We can make sure that both pure stacking(Av_Stack) and extended stacking (Av_Ext_Stack) schemes always outperform the worst single classifier. Moreover, in the case of data sets where there was a lot of difference in performance among learning algorithms, for

<Table 3> Average accuracies of different MCSs

Schemes	Data Sets								
	balance	breast	glass	heart	hepatitis	Ionosphere	iris	vote	soybean
Av_Single	79.46	94.96	57.53	80.91	81.41	83.47	94.77	93.03	89.56
Av_Bag	83.89	95.80	61.64	82.52	81.73	84.45	96.08	93.76	91.56
Av_Boost	85.20	95.17	56.85	83.01	81.73	89.50	96.08	96.64	90.78
Av_Stack	86.50	95.38	54.11	83.50	80.77	84.45	96.08	96.28	92.27
Av_StackBag	87.25	95.44	61.30	78.77	79.85	87.51	95.00	93.06	89.29
Av_StackBoost	87.32	95.61	58.64	77.08	79.78	89.69	95.22	93.68	89.39
Av_BagStack	88.32	96.00	62.28	82.57	82.62	91.60	95.00	96.44	91.88
Av_BoostStack	92.64	96.14	66.09	82.68	82.32	90.03	95.67	96.33	93.90

<Table 4> Accuracies of single classifiers and stacking schemes

Schemes	Data Sets								
	balance	breast	glass	heart	hepatitis	Ionosphere	iris	vote	soybean
C4.5	74.41	93.70	61.64	77.67	80.77	80.67	92.16	94.62	89.70
K-NN	76.30	95.80	64.38	80.58	80.77	86.55	96.08	93.24	87.98
NB	87.68	95.38	46.58	84.47	82.69	83.19	96.08	91.22	90.99
Av_Stack	86.50	95.38	54.11	83.50	80.77	84.45	96.08	96.28	92.27
Av_Ext_Stack	90.48	96.07	64.19	82.62	82.47	90.81	95.33	96.38	92.89

example, in the case of balance and heart datasets, stacking schemes showed much better performance. These results imply that stacking schemes can help stabilizing the performance by combining the different biases of learning algorithms into a single system. But in order to get this advantage from stacking schemes, we need more research on efficient meta learning methods. Finally, among four different hybrid MCSs, boosted stacking, which applies the weighted bootstrap sampling and different learning algorithms to generate base-level classifiers and then use a meta learner to combine these base-level classifiers, showed the best performance in our experiments.

5. Conclusions

This paper reviewed several methods to generate efficient multiple classifier systems, and compared their relative merits. Several hybrid MCSs such as stacked bagging, stacked boosting, bagged stacking, and boosted stacking are then newly proposed. Through some experiments with different real-world datasets, we demonstrated that the hybrid MCSs, especially extended stacking schemes including bagged stacking and boosted stacking, contribute most to error reduction. We also found that the application of different learning algorithms to generate base-level classifiers helps stabilizing the performance of MCS. But we could not confirm the superiority of meta learning to voting as classifier combiner. In order to get advantages from pure stacking and extended stacking, we need more research on meta learning methods. Based on all of these findings, it seems that the improved reliability, stability, and classification accuracy can justify the high cost of building hybrid multiple schemes.

Reference

Ali, K.M and Pazzani M.J., "Error Reduction through Learning Multiple Descriptions", Machine Learning, Vol.24, No.3, (1996) 173-206.

Blake, C. and Merz C.J., UCI Repository of Machine Learning Databases, Irvine, CA: University of California, Department of Information and Computer Science (1998).

Bauer, Eric and Ron Kohavi, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting and Variants", Machine Learning Vol.36, (1999) 105-142.

Breiman, L., "Stacked Regressions", Machine Learning, Vol.24, (1996) 49-64.

Breiman, L., "Bagging Predictors", Machine Learning. Vol.24, (1996) 123-140.

Dietterich, T.G. "An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Tress: Bagging, Boosting, and Randomization", Machine Learning, Vol.28 (2000).

Fumera G. and Roli F., "Analysis of error-reject trade-off in linearly combined multiple classifiers", Pattern Recognition, Vol.37, No.6, (2004) 1245-1265.

Hirotaka I. and Hiroyuki N., "Efficiency of Self-Generating Neural Networks Applied to Pattern Recognition," Mathematical and

Computer Modelling, Vol.38, No.11-13, (2003) 1225-1232.

Hong, Se June and Sholom M. Weiss, "Advances in Predictive Model Generation for Data Mining", IBM Research Report RC-21570 (1999).

Kittler, J. et al., "On combining classifiers, IEEE transactions on Pattern Analysis and Machine Intelligence", Vol.20, No.3, (1998) 226-239.

Mitchell, Tom. Machine Learning. New York: McGraw-Hill (1997).

Schaphire, Robert E. "Theoretical views of boosting", In Computational Learning Theory: 4th European Conference EuroCOLT '99 (1999).

Sirlantzis K. and Fairhurst M.C., "Investigation of a Novel Self-Configurable Multiple Classifier System for Character Recognition", Proceedings of ICDAR-2001, Seatle, USA, (2001) 1002-1006.

Wolpert, D. and Macready, W., "Combining Stacking with Bagging to Improve a Learning Algorithm", Technical Report, Santa Fe: Santa Fe Institute (1996).

Witten, Ian H. and Eibe, F., Data Mining: Practical Machine Learning Tools and techniques with Java Implementations, New York: Morgan Kaufman (2000).

요약

하이브리드 다중 분류기시스템

김인철*

단일 분류기보다 우수한 성능을 얻기 위해 다수의 분류기들을 결합하는 방법은 폭 넓게 이용되어 오고 있는 기술이다. 하나의 다중 분류기 시스템(MCS)를 구축하는 일은 두 가지 해결해야 할 문제들을 가지고 있다. 하나는 다양한 기반-레벨의 분류기들을 어떤 방법으로 생성하느냐 하는 것이고, 다른 하나는 이들의 예측을 어떤 방법으로 결합하느냐 하는 것이다. 본 논문에서는 기존의 다중 분류기 시스템들인 *bagging*, *boosting*, 그리고 *stacking*의 특징들을 살펴본 다음, 새로운 다중 분류기 시스템들인*stacked bagging*, *stacked boosting*, *bagged stacking*, 그리고 *boosted stacking*들을 제안한다. 이들은 기존의 다중 분류기 시스템들의 장점들을 결합한 일종의 하이브리드 다중 분류기 시스템들이다. 새로 제안한 다중 분류기 시스템들의 성능을 평가하기 위해, 본 논문에서는 UCI KDD 데이터 아카이브에서 제공되는 서로 다른 9 가지의 실세계 데이터 집합들을 이용하여 실험들을 전개하였다. 실험 결과, 본 논문에서 제안한 하이브리드 다중 분류기 시스템들, 특히 *bagged stacking*과 *boosted stacking*이 기존의 다중 분류기 시스템들에 비해 우수한 성능을 보여 주었다.

* 경기대학교 정보과학부