# COMMITTEE PATTERN CLASSIFIERS

*Yu Hen Hu , Jong-Min Park, and Thomas Knoblock*

Department of Electrical and Computer Engineering
University of Wisconsin
Madison, WI 53706
Email: hu@engr.wisc.edu

## ABSTRACT

Novel methods which combines outputs of multiple pattern classifiers to enhance the overall performance of pattern classification are presented. Specific attention is given to combination rules which are independent of the input feature vectors. Potentials and pitfalls of this so called *stack generalization method* are discussed, and experimentation using several machine learning data bases are reported.

## I. INTRODUCTION

Pattern classification is the enabling technology for speech recognition, image understanding, target recognition, and other important signal processing applications. A pattern classifier is a decision-making algorithm which determines the class label of a feature vector presented to the classifier. Based on statistical decision theory, artificial intelligence, fuzzy logic theory, and many other approaches, numerous types of pattern classifiers have been developed [3]. However, it remains an open question on which pattern classifier to use given a particular problem on hand. It is generally accepted that a universal pattern classifier which will out-perform every other pattern classifiers is unlikely to be found. The current practice is to choose one classifier which seems to perform the best for the problem on hand, or to use whatever the designer is most familiar with. The situation is analogous to the decision making process in human society where a person who knows every thing well is unlikely to be found. However, in human societies, many experts, each specialize in a sub-fields, are often summoned to form a **committee** to solve a complicate problem in a collective manner. The belief is that collective efforts can often arrive at a superior decision than by any individual expert. Motivated by this observation, recently, many researchers have been looking into the ways to combine multiple pattern classifiers' outputs to form a **committee classifier**, and hope that the overall output will be better than any individual classifier can achieve. Owing to this analogy, we will use classifier and expert interchangeably in this paper. Along this direction, a few questions will be studied in this paper:

- *Under what conditions a multiple-expert classifier will outperform any of its individual component classifier?*
- *What is the best combination method to achieve above goal?*

In the following discussion, we will denote **x** to be the current feature vector presented to a classifier, and **y**(i) as the output of the $i^{th}$ expert classifier. The output of the combined committee classifier will be denoted by **z**.

## II. COMMITTEE CLASSIFIER

Committee classifier consists of a committee of n individual pattern classifiers. There outputs, denoted by $\{\mathbf{y}(i);\ 1 \le i \le n\}$ are to be combined, linearly or non-linearly, via a set of combination rules, to form the final output, **z**. For classification problem, the outputs **y**(i) and **z** are c by 1 vectors with a "1" in an $k^{th}$ entry indicating the classifier decides that the input feature vector **x** belong to the $k^{th}$ class. Usually, one would allow only one element to be 1 and the rest should remain at 0.

We define a committee classifier as one that its combination rules are not a function of individual feature vector **x**. In other words, *in a committee classifier, the combination rules are determined during the training phase, and are not subject to change during testing phase when particular* ***x*** *is presented.*

There are several empirical studies of combining multiple classifiers reported in literature [1], [2], [4], [5], [6], [7], [8], [9], [10]. Several specific combination rules are now discussed:

### *II.1 Bayes Combination Rules*

In a Bayes combination rule, it is often assumed that the output of the i-th member classifier, $\mathbf{y}(\mathbf{x},i)$, is an estimate of the posterior probability given input $\mathbf{x}$, and classifier structure i, $\mathbf{P}\{y|\mathbf{x}, i\}$. Under this assumption, we must allow the entries of $\mathbf{y}(\mathbf{x},i)$ and $\mathbf{z}$ to be a real number varying between 0 and 1, and require the sum of these entries of each of these vectors to be equal to unity. The Bayes combination rule then utilizes the *Bayes' rule* to derive the overall estimate

$$\mathbf{z}(\mathbf{x}) = \mathbf{P}\{y \mid \mathbf{x}\} = \sum_{i=1}^{n} \mathbf{P}\{y|\mathbf{x}, i\}\, \mathbf{P}\{i|\mathbf{x}\}$$

$$= \sum_{i=1}^{n} \mathbf{y}(\mathbf{x},i)\ \mathbf{w}(i,\mathbf{x}) \qquad (1)$$

Clearly, we have $\mathbf{w}(i,\mathbf{x}) = \mathbf{P}\{i|\mathbf{x}\}$ where $\mathbf{P}\{i|\mathbf{x}\}$ is the conditional prior probability of the classification performance of each individual classifier overall the entire feature space. One may also consider different weights for different classes to further enhance the performance.

The dependency of the weight on the input feature vector $\mathbf{x}$ leads to the development of the *mixture of expert* modular network architecture. While there are many approaches to derive a good estimate of $\mathbf{w}(i,\mathbf{x})$, we propose the following approach: First, partition the training data set into regions using techniques such as clustering. Within each cluster, the classification rate (on the feature vectors within that cluster) of each classifier will be calculated. Then, they will be normalized such that $\sum \mathbf{w}(i,\mathbf{x}) = 1$.

### *II.2 Linear Weighted Combination Rules*

A second approach is based on a model of $\mathbf{y}(\mathbf{x},i) = \mathbf{P}\{y|\mathbf{x}, i\} + \varepsilon(i|\mathbf{x})$ where $\varepsilon(i|\mathbf{x})$ are random estimation errors with zero mean and variance $\sigma_i^2(\mathbf{x})$. Then the objective is to find a set of weights $\{\mathbf{w}(i);\ 1 \le i \le n\}$ such that the variance of the overall linear estimate $\|\mathbf{z}(\mathbf{x}) - \sum_{i=1}^{n} \mathbf{y}(\mathbf{x},i)\mathbf{w}(\mathbf{x},i)\|^2$ is minimized. This *minimum variance estimate* so obtained is

$$\mathbf{w}(\mathbf{x},i) = \frac{1/\sigma_i^2(\mathbf{x})}{\sum [1/\sigma_j^2(\mathbf{x})]}. \qquad (2)$$

In other words, the weights are inversely proportional to the variance of the estimate. To apply this method, one must estimate the local variance of $\sigma_i^2(\mathbf{x})$ for each expert classifier. Similar to the case of Baysian combination rule, we propose to first cluster the feature vectors $\mathbf{x}$ into individual clusters, and calculate $\sigma_i^2(\mathbf{x})$ within each cluster for each classifier.

## III NONLINEAR COMBINATION RULES

Nonlinear combination rules can be regarded as a general meta-classifier designed to classify a concatenated feature vector $\mathbf{y}(\mathbf{x}) = [\mathbf{y}(\mathbf{x},1)\ \mathbf{y}(\mathbf{x},2)\ \ldots\ \mathbf{y}(\mathbf{x},n)]$. Any known classifier structures, such as MAP (maximum a posterior probability), kNN (k nearest neighbors), SOM (self-organization map), decision trees (e.g. ID3), can be applied to serve this purpose. The question is, is there any way to predict how the committee classifier performs compared to individual member classifiers? Let us examine a special case below when both inputs (feature vectors) and outputs of each classifier are discrete value in {0, 1}. In this case, there are only $2^k$ different input combinations where k is the feature vector dimension.

| x1 | x2 | x3 | y1 | y2 | y3 | y4 | y(desired) |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figure 1. Examples illustrating aliasing effect of a committee classifier**

In figure 1, x1, x2, and x3 are features. y is the desired mapping and y1-y4 are 4 imperfect member classifiers. Shaded cells indicate misclassification of the corresponding classifier. The question now is: Given y1, y2, y3, y4, can a meta-classifier (combination rules) be defined so that it gives an output which is the same as y? Here is an incomplete 4 Boolean variable minimization problem, and one of the solution is: y = y2&y4 + y1&y4 + y1 &y2. In other words, combining y1, y2, and y4, the committee classifier is able to yield 100% classification rate on this training data set — a performance better than any individual classifier. On the other hand, if there are only y1, y2, y3 are available, note that when (y1,y2,y3) = 010, and 101, both of them appear twice with different values of y. Thus, one can choose only one of the values. This implies that the maximum classification rate will be at most 6/8 which is no better than either y1 or y2 alone. This phenomenon of having different target values associated to the same classifier output combination is called aliases. The output of multiple classifiers can be used as an induced feature vector to achieve perfect classification rate on training set data if aliases does not occur.

One way to ensure alias will never occur is to use an extended committee classifier architecture as shown in Figure 2. With the configuration illustrated in Figure 2, the committee classifier not only will use all expert classifiers' output as its feature vector, but it will also take the original feature vector x. As such, an extended feature vector [x y1 y2 . . . yk] is used.

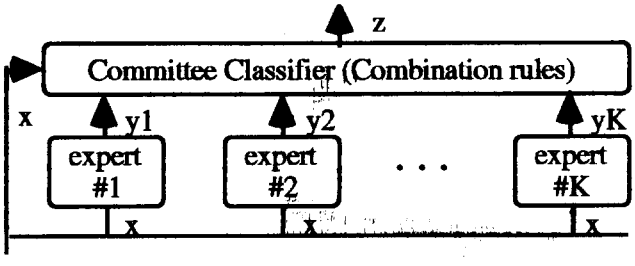


**Figure 2. Alias-Free Committee Classifier with Direct Feature Feed-Through**

Since original classification problem is assumed to be free from alias, so will this new classification problem with the extended feature space.

## IV. EXPERIMENTATION

We have employed the four data sets from machine learning database at UCI. They are: A. credit card applications, B. breast cancer diagnosis, C. DNA Promoter sequence recognition, and D. poisonous mushroom identification. Each data file is randomly partitioned into three parts. A *three-way cross-validation* procedure is adopted to better estimate the generalization error: Each method is applied three times (trials) to each data file. In each trial, two of the three parts are used as training data, and the third as the testing data. After three trials, each of the three parts of the original data file will be tested exactly once. The testing error rates of the three trials then are averaged to yield the overall classification rate of a particular classification method on a given data set.

Four expert classifiers are used in this experiment: a 3-nearest neighbor (3NN) classifier, a maximum likelihood (ML) classifier, a Learning vector quantization (LVQ 2.1) classifier, and a multi-layer perceptron (MLP) classifier. In developing the ML classifier, feature vectors in each class is assumed to have a normal distribution. Thus, it effects a linear classifier. With the MLP classifier, a 2-layer, fully connected configuration (one hidden layer) is used, with 10 hidden units - a number assigned arbitrarily. Since our objective here is not to compare performance of individual classifiers, sub-optimal implementation of these classifiers should not prevent us from comparing results between the committee classifier to the best of the individual classifiers. Each of these four classifiers will be used as the committee classifier classifying not only the output of the member classifiers, but also the original feature vector to facilitate aliases-free classification.

All but the LVQ algorithm are implemented with Matlab (v.4.2c) m-files, tested on a HP workstation. The LVQ algorithm is implemented by the SOM research group of the University of Helsinki, and is available at *ftp://cochlea.hut.fi/pub/*.

Note that for each data set and each classification method, there are actually three different expert classifiers developed - each developed on one of the three different training

data set. In this paper, we will simply treat them as three independent trials, and will not be concerned with how to combine these three different classifiers trained with the same data set (on different partitions). Instead, for each partition, we will compare the potential performance gain by combining the output of several different classifiers, trained on the same partitions.

For this purpose, first we construct an induced feature vector which consists of the outputs of each of the four classifiers. Then we develop a committee classifier to classify these extended feature vectors using each of the four types of classifiers (3-NN, ML, LVQ, MLP). Again, three trials are performed on each of the three different partitions of each data set, and the results are reported below:

**Table 1. Classification error rates of committee classifiers with outputs of member classifiers only**

| | Voting | ML | 3-NN | LVQ 3.0 |
|---|---|---|---|---|
| Cancer | 4.98% | 2.49% | 3.64% | 4.02% |
| Card | 19.18% | 19.18% | 19.77% | 19.18% |
| Gene | 21.98% | 26.86% | 22.66% | 14.88% |
| Heart | 22.03% | 22.09% | 22.90% | 22.03% |

Next, we augment the induced feature vector by the original feature vector so that no aliases may occur. Then we repeat above experiment using 3-way cross-validation. The results are summarized below:

**Table 2. Classification error rate of extended committee classifiers**

| | Voting | ML | 3-NN | LVQ3.0 |
|---|---|---|---|---|
| Cancer | 4.98% | 5.94% | 2.30% | 4.02% |
| Card | 19.18% | 18.22% | 19.57% | 18.60% |
| Gene | 21.98% | 36.62% | 25.68% | 14.71% |
| Heart | 22.03% | 23.04% | 22.46% | 20.87% |

From above two tables, we observe that compared to simple majority voting, the committee machine approach, at least with this experiment, does not significantly improve the classification performance in general. Among the three different classifiers, LVQ 3.0 seems consistently out-perform the voting method, while other two classifiers gives mixed results. Compare the committee method, and the extended committee method, where original features are used, the results are mixed. Our preliminary explanation is that the additional dimension causes the ML or 3-NN based committee classifier confused.

## REFERENCES


[1] Battiti, R., and A. M. Colla, "Democracy in neural nets: voting schemes for classification," *Neural Networks*, vol. 7, no. 4, pp. 691-709, 1994.

[2] Drucker, H., C. Cortes, L. Jackel, Y. LeCun, and V. Vapnik, "Boosting and other ensemble methods," *Neural Computation*, vol. 6, no. 6, pp. 1289-1301, 1994.

[3] Duda, R. O., and P. E. Hart, *Pattern classification and scene analysis*. New York: Wiley, 1973.

[4] Hansen, L. K., and P. Salamon, "Neural network ensembles," *IEEE Trans. on PAMI*, vol. 12, no. 10, pp. 993-1001, 1990.

[5] Ho, T. K., J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *IEEE Trans. on PAMI*, vol. 16, no. 1, pp. 66-76, 1994.

[6] Meir, R., "Bias, variance, and the combination of estiamtors: the case of least linear squares," in *Advances in Neural Information Processing Systems 7*, Ed(s)., Cambridge, MA: MIT Press, 1995,

[7] Perrone, M. P., and L. N. Cooper, "When networks disagree: ensemble method for neural netowrks," in *Neural Netowrks for Speech and Image Processing*, R. J. Mammone, Ed(s)., Chapman-Hall, 1993,

[8] Twomey, J. M., and A. E. Smith, "Committee networks by resampling," in *Intelligent Engineering Systems Through Artificial Neural Networks*, C. H. Dagli et al, Ed(s)., ASME Press, 1995, pp. 153-158.

[9] Wolpert, D. H., "Stacked Generalization," *Neural Networks*, vol. 5, no. 2, pp. 241-259, 1992.

[10] Wolpert, D. H., "Combining generalizers using partitions of the learning set," in *1992 Lectures in Complex Systems*, Addison Wesley, 1993, pp. 489-500.