Pergamon

Neural Networks, Vol. 7, No. 4, pp. 691–707, 1994

0893-6080/94 $6.00 + .00

0893-6080(94)E0015-D

CONTRIBUTED ARTICLE

Democracy in Neural Nets: Voting Schemes for Classification

ROBERTO BATTITI[1] AND ANNA MARIA COLLA[2]

[1]Università di Trento and [2]Elsag Bailey

(*Received* 3 *December* 1992; *revised and accepted* 24 *June* 1993)

Abstract—*In this paper we discuss some possible ways to combine the outputs of a set of neural network classifiers to reach a combined decision with a higher performance, in terms of lower rejection rates and/or better accuracy rates. The methods considered range from the requirement of a complete agreement among the individual classifications to election schemes based on the distribution of votes collected by the different classes. In addition, the rejection rules based on the different output classes can be complemented by rules that also consider the information in the individual output vectors, with the possibility of using threshold requirements and that of averaging the different vectors. Although the Bayesian framework and some probabilistic assumptions provide useful indications about the potential advantage of different combination schemes, the combined performance ultimately depends on the joint probability distribution of the outputs, and it can be estimated by joining the results of different nets on the same test set. The combination methods are very flexible, they permit a straightforward cooperation of neural and traditional recognizers, and they are appropriate in a development environment where experiments are performed with different kinds of nets and features for a selected application. From our experiments in the field of handwritten digit recognition (up to a total of more than* 50,000 *characters), we found that the use of a small number of nets (two to three) with a sufficiently large uncorrelation in their mistakes reaches a combined performance that is significantly higher than the best obtainable from the individual nets, with a negligible effort after starting from a pool of networks produced in the development phase of an application. In particular, for a real-world OCR application, the best accuracy increase is about half the increase in the rejection rate, so that accuracies of the order of* 99.5% *can be reached by rejecting less than* 5% *of the patterns. This performance is significant for real applications.*

Keywords—Modular neural networks, Multilayer perceptron, Bayesian classification, Accuracy–rejection trade-off, Optical character recognition.

1. INTRODUCTION

In many practical applications of recognition systems, the desired accuracy rate can be obtained only by rejecting a fraction of the patterns. It is evident that the system should reject the patterns that have a high probability of being wrongly classified because these are the patterns that degrade the accuracy. Different rejection criteria have been used for recognition systems based on neural nets (LeCun et al., 1989) for an application to zip code recognition. In this paper we analyze the classification of independent networks and consider the *disagreement* between them as the *symptom* of an uncertain classification. Each net is assigned a vote. Different voting schemes position the combined multiple-network system on different points of the rejection–accuracy plane, and the final selection of the best system is based on the requirements of the specific application.

A theoretical basis of these rejection schemes can be obtained from Bayes' decision theory and from the results stating that the outputs of multilayer perceptron neural nets approximate posterior probabilities for the different classes (Ruck et al., 1990; Wan, 1990). For all recognition systems, including neural nets, the probability of an accurate classification cannot be larger than the Bayesian limit.

Let $P(\omega_i)$ be the a priori probability for the ith class ($i = 1, \ldots, C$), $p(\mathbf{x}|\omega_i)$ be the *state-conditional* probability density. Then the *posterior* probability $P(\omega_i|\mathbf{x})$ can be computed from $p(\mathbf{x}|\omega_i)$ by Bayes rule:

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{p(\mathbf{x})},$$

where the global probability density $p(\mathbf{x})$ can be obtained as:

$$p(\mathbf{x}) = \sum_{i=1}^{C} p(\mathbf{x}|\omega_i)P(\omega_i) \qquad (1)$$

Acknowledgement: We acknowledge useful discussions with Dr. Pietro Pedrazzi, Elsag Bailey, about the LVQ algorithm.

Requests for reprints should be sent to Roberto Battiti, Dipartimento di Matematica Università di Trento, 38050 Povo, Trento, Italy.

Considering a task with C classes, and a classifier partitioning the input space in C *decision regions* $\mathcal{R}_i$, so that the patterns in region $\mathcal{R}_i$ are associated to class ω_i, the probability density of a correct classification at point $\mathbf{x}$ in region $\mathcal{R}_i$ is $p(\mathbf{x}|\omega_i)P(\omega_i)$, the term in the sum (1) that belongs to class ω_i. After integrating over the different regions and summing the contributions (Duda & Hart, 1973) the global probability of a correct recognition is:

$$P(\text{correct}) = \sum_{i=1}^{C} \int_{\mathcal{R}_i} p(\mathbf{x}|\omega_i)P(\omega_i)\, d\mathbf{x} \qquad (2)$$

The Bayes classifier maximizes this probability by choosing the regions so that the integrands are maximum, that is, by assigning a pattern to the class ω_i that maximizes $p(\mathbf{x}|\omega_i)P(\omega_i)$. By Bayes' rule $p(\mathbf{x}|\omega_i)P(\omega_i)$ is proportional to $p(\omega_i|\mathbf{x})$, the *posterior probability* of class ω_i given the input pattern $\mathbf{x}$ so that one may as well maximize the latter quantity over i.

The Bayes' theoretical performance can be reduced by erroneous estimates of the probabilities from a finite number of samples (i.e., generalization errors) and by mistakes in the data acquisition and/or preprocessing phase. Considering, for example, a character recognition task, it is possible to choose appropriate input features so that the probability densities $p(\text{character}|\mathbf{x})$ are almost nonoverlapping (for each pattern $\mathbf{x}$ there is a clear-cut winning class), and therefore the Bayesian limit is close to one. Nonetheless, the performance of a recognition system can degrade in tests with a different writer or with different writing styles (*generalization*) or with wrongly segmented or wrongly normalized characters (*preprocessing*). Particularly in the last case, diagnosing the problem can be preferable to providing an unreliable classification: a complete recognition system could in this case amend the segmentation or normalization.

In the following, first we introduce the *rejection–accuracy* plane as a tool for discussing the trade-off between accuracy of classification and fraction of accepted patterns, and derive the criterion for the optimal (Bayesian) rejection (Section 2). Then we discuss different methods to combine the outputs of a *team* of classifiers, ranging from probabilistic schemes (Section 3), to *unison* or *majority* schemes, based on the agreement of all or the majority of the votes in the poll, respectively (Sections 4 and 5). Finally, we review some rules for using the analog *output activation values* of the networks, in addition to the *winner-takes-all* classification, for the final decision (Section 6), and discuss the effects of *averaging* the outputs of several nets (Section 8).

For each of the above topics, a set of experiments has been executed to complement the theoretical analysis. In particular, we present a test of the Bayesian criterion for a multilayer perceptron neural net trained on a Gaussian mixture distribution in Section 2.2 and three sets of tests in the handwritten optical character recognition (OCR) domain, on training and test sets of increasing size and interest for real-world applications. The OCR experiments test the unison (Sections 4.4 and 4.5) and majority schemes (Section 5.1), the use of previously defined schemes plus the analog output values for networks using the same set of features, and the back propagation training algorithm (Section 7) and the use of network averaging (Section 8).

2. THE ACCURACY–REJECTION COMPROMISE

Let us consider classification systems composed of multiple networks. To avoid unnecessary complications we use the following notation for the probabilities associated to events: $p(\text{event}_1, \text{event}_2, \ldots)$ is the probability that event_1 and event_2 and ... happen. For example, $p(w_1, w_2, \text{equal})$ is the probability that the responses of both classifiers 1 and 2 are wrong and that the output classes are equal. The specific $p(\ )$ function is uniquely identified by its arguments.

In the following, we consider an abstract description of a classifier as a system that processes a vector $\mathbf{x}$ of inputs and provides both a decision (i.e., a class) and a binary value about the confidence in the classification. If the confidence *flag* is set to zero (uncertain response), the pattern is *rejected* by the classifier. The performance of a classifier can be described by a point in the *rejection–accuracy plane* (briefly *R-A*), that is, by its probability of an accurate response, given that the input pattern is accepted: $A = p(\text{correct}|\text{accept})$, and by its probability of rejection $R = p(\text{reject})$. Note that the *accuracy* is always conditional to the acceptance of the pattern. If several classifiers are involved, we define $R_i \equiv p(\text{reject}_i)$, and $A_i \equiv p(\text{correct}_i|\text{accept}_i)$, for the ith classifier.

In general, the *R-A* values depend on the internal parameters of the classifier. By varying these parameters one obtains the accuracy as a function of the rejection rate: $A(R)$, a function that depends on the specific rejection scheme. The $A(R)$ function is increasing with R if a growing fraction of the accepted cases consists of correctly classified patterns.

The *R-A* coordinates introduce a partial ordering of different classifiers: classifier X is better than classifier Y if it has both a greater accuracy and a smaller rejection. In the other cases (for example if X has greater accuracy but also greater rejection than Y), the preference is decided by a *compromise* between the two requirements of high accuracy and low rejection. Introducing a parameter λ regulating the relative importance of the two requirements, the optimal classifier for a given application can be defined as the one that maximizes $\mathcal{U} \stackrel{\text{def}}{=} A - \lambda R$ with $\lambda \geq 0$. In Figure 1 (left) we illustrate graphically how the gradient of the compro-

mise function $\mathcal{U}$, equal to $(-\lambda, 1)$, introduces a complete ordering of the classifiers, apart from ties. Other performance criteria can be based on the simultaneous satisfaction of two inequalities of the type: $A \geq A_{\min}$ and $R \leq R_{\max}$ (Figure 1, right).

2.1. Optimal (Bayesian) Rejection

If an application needs a higher accuracy than the Bayesian limit of eqn (2), one needs to reject the patterns that belong to a portion of the input space. Now, at a given point $\mathbf{x}$ in the input space, the relative proportion of patterns belonging to the different classes is $p(\omega_i|\mathbf{x})$, and the optimal fraction of correctly classified patterns is equal to $\max_i p(\omega_i|\mathbf{x})$. As the fraction of rejected patterns increases, it is natural that the patterns to be rejected first are those with a low probability for the winning class (i.e., for the class with the maximum probability). These patterns are in regions of the input space where the distributions for the different classes are overlapped.

Let us consider a threshold criterion for accepting patterns such that the threshold T_{accept} is constant (i.e., independent of $\mathbf{x}$). In general, at a given rejection rate, the best accuracy can be obtained by accepting only the patterns for which the maximum probability $p(\omega_i|\mathbf{x})$ is greater than or equal to a threshold T_{accept}. The acceptance criterion for $\mathbf{x}$ can be reformulated as:

$$\max_i p(\omega_i|\mathbf{x}) \geq T_{\text{accept}} \quad \text{or}$$

$$\frac{\max_i[P(\omega_i)p(\mathbf{x}|\omega_i)]}{p(\mathbf{x})} \geq T_{\text{accept}} \tag{3}$$

where Bayes' rule is used to transform one relation into the other, and $p(\mathbf{x})$ is equal to $\sum_{k=1}^{C} P(\omega_k)p(\mathbf{x}|\omega_k)$, the cumulative probability density.

By increasing T_{accept}, larger portions of the input space are assigned to the rejected region $\mathcal{R}$. A short demonstration of the optimality of the threshold-based rejection of eqn (3) is presented in Appendix A.

When the threshold T_{accept} is increased so that new points are rejected, a first-order approximation to ratio $\Delta A/\Delta R$ is given by:

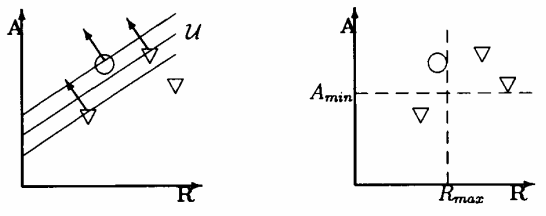


FIGURE 1. Selection of the optimal classifier (point ○ in the rejection–accuracy plane), based on: (*left*) maximizing $\mathcal{U} \stackrel{\text{def}}{=} A - \lambda R$ with $\lambda \geq 0$ and (*right*) applying thresholds to A and R.

$$\Delta A/\Delta R \equiv \frac{\Delta p(\text{correct}|\text{accept})}{\Delta p(\text{reject})} \approx \frac{1}{p(\text{accept})}\left[\frac{\int_{\mathcal{A}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x}}{\int_{\mathcal{A}} p(\mathbf{x})\,d\mathbf{x}} - \frac{\int_{\Delta\mathcal{R}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x}}{\int_{\Delta\mathcal{R}} p(\mathbf{x})\,d\mathbf{x}}\right] \tag{4}$$

where $\mathcal{A}$ is the currently accepted region and $\Delta\mathcal{R}$ is the new portion of the input space that is going to be included in the rejection region (see Appendix A for additional considerations).

2.2. Neural Nets to Realize Optimal Rejection

Recently, it has been demonstrated that multilayer perceptrons (MLPs) can be used to estimate the probability densities $p(\omega_i|\mathbf{x})$ from a finite set of examples (Ruck et al. 1990; Wan, 1990). By using this result and the optimality of the threshold-based rejection criterion of eqn (3) one should obtain a close approximation to the optimal $A(R)$ curve by using thresholds on the network output values.

Here we present a detailed example by calculating the optimal rejection–accuracy values and comparing them to those obtained with thresholds on the maximum output value of a MLP classifier. We consider a case in which the pattern distribution is a mixture of Gaussian densities. The samples are assumed to be generated by selecting a prototype ω_i with probability $P(\omega_i)$ and then selecting a pattern $\mathbf{x}$ with a normal (Gaussian) probability $p(\mathbf{x}|\omega_i)$. A general multivariate normal density in d dimensions can be written as:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m})'\Sigma^{-1}(\mathbf{x}-\mathbf{m})\right] \tag{5}$$

where $\mathbf{m}$ is the *mean vector* ($\mathbf{m} = E[\mathbf{x}]$) and Σ is the *covariance matrix* ($\Sigma = E[(\mathbf{x}-\mathbf{m})(\mathbf{x}-\mathbf{m})']$).

We now present a two-class discrimination experiment derived from Burrascano (1991), where each class has the same probability and is described by a mixture of Gaussian densities. The first class is described by a Gaussian distribution elongated along the y axis, the second one is given by the mixture of two Gaussians displaced in the x direction. After introducing the one-dimensional distribution:

$$N(v, \mu, \sigma) = (2\pi\sigma^2)^{-1/2}\exp\left[-\frac{(v-\mu)^2}{2\sigma^2}\right] \tag{6}$$

that is a Gaussian with mean μ and variance σ^2, the probability densities for classes 1 and 2 are:

$$p(x, y|\text{class} = 1) = N(x, 0, \sigma)N(y, \mu_y, 2\sigma)$$

$$p(x, y|\text{class} = 2) = \tfrac{1}{2}[N(x, \mu_x, \sigma) + N(x, -\mu_x, \sigma)]N(y, -\mu_y, \sigma).$$

The values for the parameters are $\sigma = 0.1$, $\mu_x = 1.188\sigma$, and $\mu_y = 2.325\sigma$. The distributions for the two classes are illustrated in Figure 2.

The degree of certainty in the classification (i.e., the value $\max_i p(\omega_i|\mathbf{x})$ as a function of the input coordinates) and the Bayes optimal rejection regions defined by eqn (3) are shown in Figure 3, where the contours correspond to increasing values of the T_{accept} threshold.

The Bayes *R-A* values have been calculated by a numerical integration on the accepted regions determined by four values of the threshold T_{accept}. This optimal result[1] is then compared with the one obtained by a multilayer perceptron neural network, with a rejection criterion based on a threshold on the maximum acti-

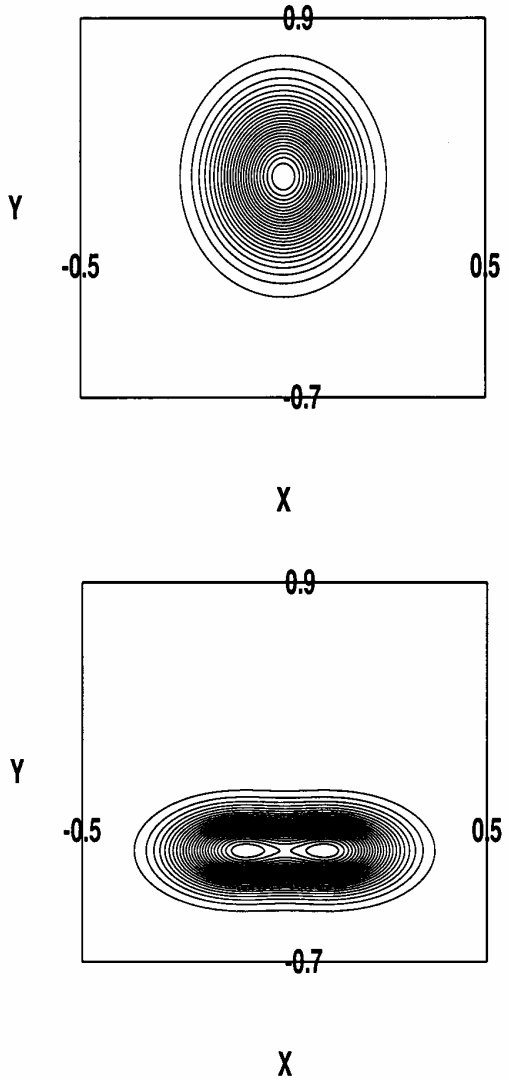


FIGURE 2. Test with Gaussian mixtures: distribution for class 1 [$p(\omega_1)p(\mathbf{x}|\omega_1)$, above] and class 2 [$p(\omega_2)p(\mathbf{x}|\omega_2)$, below]. Class 1 is described by a single Gaussian, class 2 by the superposition of two Gaussians symmetrically displaced along the x axis. Equally spaced contours with $\Delta z = 0.2$. The view is taken in the X − Y region [−0.5, 0.5] × [−0.7, 0.9].

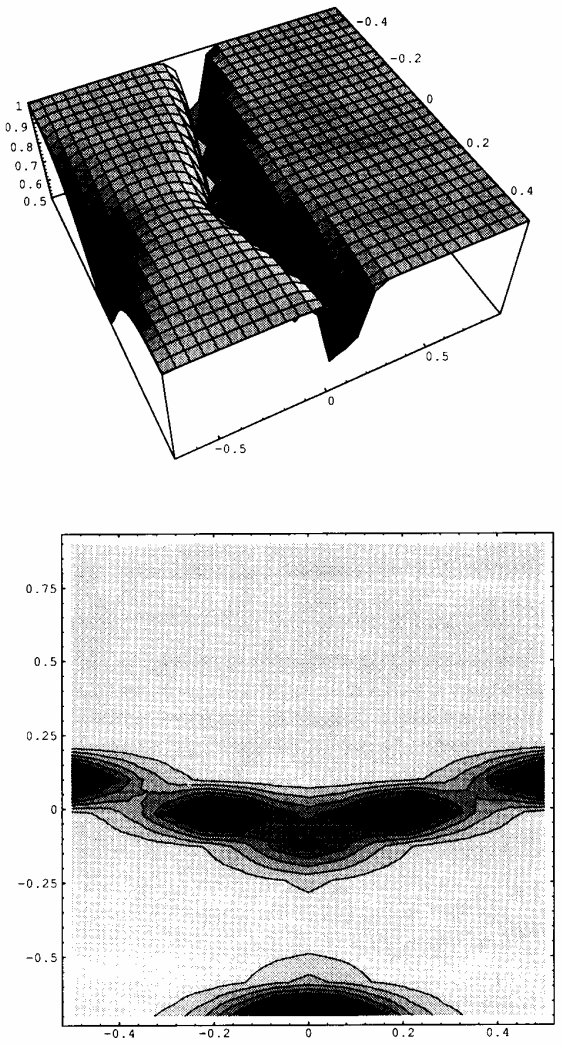
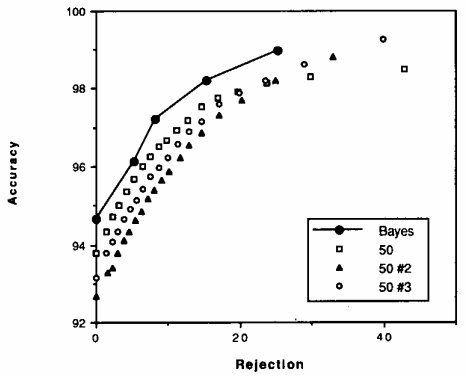


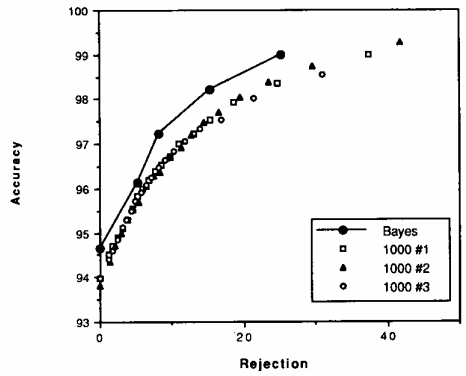
FIGURE 3. Gaussian mixture example: three-dimensional plot of $\max_i p(\omega_i|\mathbf{x})$ (top) and border of the rejection region for increasing values of the threshold T_{accept}, equally spaced from 0.5 to 0.95 (bottom). Dark regions correspond to uncertain classification, that is, low values of $\max_i p(\omega_i|\mathbf{x})$. The view is taken in the X-Y region [−0.5, 0.5] × [−0.7, 0.9].

[1] In detail, after introducing the function $\theta(x)$ that is equal to 1 if $x \geq 0$ and 0 otherwise, and remembering that $P(1) = P(2)$ and that our distributions are symmetrical with respect to the vertical axis, we computed the integrals: $p(\text{accept}) = \int_{x=0}^{x=+\infty}\int_{y=-\infty}^{y=+\infty} \theta\{\max[p(x, y|1), p(x, y|2)]/[p(x, y|1)\int_{x=0}^{x=+\infty} + p(x, y|2)\int_{x=0}^{x=+\infty}] - T_{\text{accept}}\}[p(x, y|1) + p(x, y|2)]\, dx\, dy$ and $p(\text{accept, corr}) = \int_{x=0}^{x=+\infty}\int_{y=-\infty}^{y=+\infty} \theta\{\max[p(x, y|1), p(x, y|2)]/[p(x, y|1)\int_{x=0}^{x=+\infty} + p(x, y|2)\int_{x=0}^{x=+\infty}] - T_{\text{accept}}\}\max[p(x, y|1), p(x, y|2)]\, dx\, dy$. The numerical integration has been executed with the *Mathematica*© software with an accuracy of at least three digits.

vation value. This method is justified by the interpretation of the output values as estimates of the posterior probabilities for the different classes and by the use of eqn (3). The network has the architecture 2-3-2, with the two output units coding for the two classes: (1, 0) for class one and (0, 1) for class two. The network is initialized with small random weight values in $[-0.5, 0.5]$ and trained for 50,000 *on-line* iterations (*learning rate* = 0.2). The trained network is then tested on a disjoint set with 10,000 samples extracted from the distribution. In Figure 4 we compare the $A(R)$ curve obtained in the case of a learning set composed of 50 and 1000 samples, respectively, for three different values of the random seed. The learning sets for each test are randomly extracted from the statistical distribution of the two classes. In the case of 1000 training samples, the $A(R)$ curves have a small standard deviation and are close to the optimal (Bayesian) curve.



A



B

FIGURE 4. Multilayer perceptron for the Gaussian mixture discrimination. $A(R)$ curves for three learning sessions (different initialization and different patterns extracted from the distribution), based on 50 (A) and 1000 examples (B). Learning with on-line back propagation (50,000 pattern presentations, learning rate = 0.2). The $A(R)$ curve with 1000 examples is close to the Bayes optimal curve.

3. PROBABILISTIC COMBINATION OF TWO CLASSIFIERS

If more classifiers are available for the same task, possibly with different input features extracted from the raw data, one can build *team classifiers,* in analogy with the human way of reaching a pondered decision after consulting a team of experts. In the following we consider teams composed of *independently* trained networks and investigate some of the possible *voting schemes* in order to combine the individual classifications. By varying the members of the team and the voting scheme, one obtains a cloud of points in the R-A plane. If the performance function is given by $\mathcal{U}$ (see Figure 1), it is possible to limit the consideration to the upper-left vertices of the convex hull of this set.

We consider now a *team classifier* with a *probabilistic selection scheme,* so that the response of the team becomes equal to the response of the ith member with probability P_i. In particular, to simplify the notation, we consider the case of two classifiers characterized by parameters (R_1, A_1), (R_2, A_2). Without loss of generality[2] we further assume that $A_2 > A_1$ and $R_2 > R_1$. By varying P_1 (clearly $P_2 = 1 - P_1$) one obtains a series of classifiers with R-A parameters $(R_{\text{team}}, A_{\text{team}})$ given by the following equations:

$$R_{\text{team}} \equiv p(\text{reject}_{\text{team}}) = P_1 R_1 + P_2 R_2 \tag{7}$$

$$A_{\text{team}} \equiv \frac{p(\text{correct}_{\text{team}}, \text{accept}_{\text{team}})}{p(\text{accept}_{\text{team}})} = \frac{P_1(1 - R_1)A_1 + P_2(1 - R_2)A_2}{1 - P_1 R_1 - P_2 R_2}. \tag{8}$$

The equations are obtained from the definition of conditional probability and the sum of probabilities for disjoint events. The admissible R-A values from the combination of two classifiers with parameters equal to $(R_1 = 0.0, A_1 = 0.5)$, $(R_2 = 0.5, A_2 = 0.9)$ are shown in Figure 5.

The function $A(R)$ has a positive second derivative in the intermediate region and therefore the optimal points correspond to one of the original classifiers in the case of the compromise performance function given by $\mathcal{U}$. On the contrary, the combination can be useful if the required accuracy is between A_1 and A_2 and the allowed rejection is between R_1 and R_2. In this case, with a negligible overhead for the probabilistic choice, the combined classifier can satisfy the requirements.

It is interesting to consider the difference $\delta(P_1)$ between the A value on the segment joining the two points of the original classifiers, and the A value for the combined classifier, as a function of the probability for selecting the first classifier:

[2] The other cases are either symmetric or not interesting because one of them is always better considering $\mathcal{U}$.

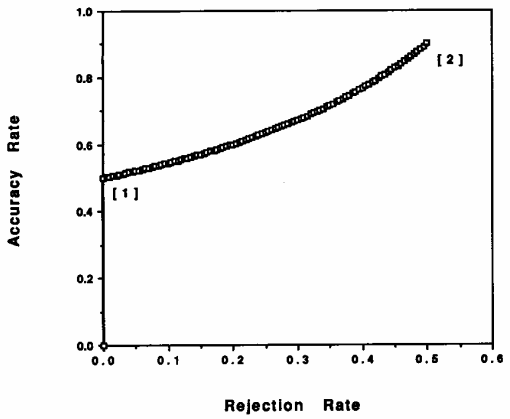


FIGURE 5. Admissible rejection–accuracy values for the probabilistic combination of two classifiers at points [1] and [2] in the *R-A* plane.

$$\delta(P_1) \equiv P_1A_1 + P_2A_2 - \left[\frac{P_1(1 - R_1)A_1 + P_2(1 - R_2)A_2}{1 - P_1R_1 - P_2R_2}\right]. \quad (9)$$

By introducing the quantities $\Delta A \equiv (A_2 - A_1)$ and $\Delta R \equiv (R_2 - R_1)$ we can transform the above expression into:

$$\delta(P_1) = \Delta A\left(\frac{P_1P_2\Delta R}{1 - P_1R_1 - P_2R_2}\right). \quad (10)$$

The denominator $(1 - P_1R_1 - P_2R_2)$ can also be written in the form $[P_1(1 - R_1) + P_2(1 - R_2)]$. Because ΔA and ΔR have the same sign, the function $\delta(P_1)$ is positive for $P_1 \in [0, 1]$ and it is maximized for:

$$P_1 = \frac{\sqrt{(1 - R_1)(1 - R_2)} - (1 - R_2)}{(R_2 - R_1)} \quad (11)$$

For the case illustrated in Figure 5, the maximum value $\delta = 0.068 = (0.17\Delta A)$ is reached for $P_1 = 0.414$: this is the maximum error if the linear approximation $(A = P_1A_1 + P_2A_2)$ is used. For a realistic combination of OCR classifiers (see the following sections) with $R_1 = 0.0$, $A_1 = 0.96$, $R_2 = 0.1$, $A_2 = 0.998$ the maximum error $\delta = 0.001 = (0.026\Delta A)$ is reached for $P_1 = 0.486$. In this case, the difference with the linear approximation is negligible in practice.

In the example presented above only one classifier was consulted for a given input pattern (after the probabilistic choice), in the following part we present strategies that consider the output values of several classifiers for each classification.

4. THE UNISON SCHEME FOR MULTI-NET CLASSIFIERS

As a starting hypothesis, let us assume that the individual classifiers do not have any rejection flag. The rejection is based only on the comparison of the outputs provided by the set of classifiers. Let us define as *output class* or *response* of a MLP neural network the index of the output unit featuring the maximum output for a given input pattern.

The basic assumption of our work is the following:

> ABA (accuracy by agreement): *if more networks agree on their classification, the chance that the classification is accurate increases.*

In other words, an inaccurate classification is signaled (with a high probability) by the disagreement between the responses of different networks. The assumption is motivated if the different networks tend to produce uncorrelated mistakes. If a pattern of class X is wrongly classified as belonging to class Y, the chance that other networks wrongly classify it as belonging to a different class Z must be substantial. This can occur if there are many possible output classes with complex decision boundaries, for example, if the various networks have different input features, architectures, initializations, or learning algorithms.

In the Introduction we hypothesized two causes for wrong classifications: mistakes during the feature extraction phase and generalization errors (plus errors caused by the superposition of the true probability distributions if these are not separated). Now, if a wrong classification is caused by a mistake in the feature extraction phase, the network output will tend to be random and ABA is justified. If a wrong classification is caused by a pattern that is distant from the examples presented during the training stage, and if the class boundaries are complex, there will be a high probability that this pattern will end up in different decision regions for different networks.

Motivated by the above reasons, we consider ABA as a working hypothesis for our derivations, noting that specific recognition tasks will require some modifications. For example, in character recognition, the confusion between digits 3 and 9 tends to be higher than the one between 3 and 1, at least for features extracted from the original image by local receptive fields. Nonetheless, ABA is justified if there are more than two confusion classes (for example 3, 9, 5, 6), if the features for different nets are different, or if many errors are caused by preprocessing mistakes.

4.1. Common Consent (Unison) Rules

Let us start by considering two networks. We are interested in four possible outcomes of the classification. The network responses can be:

i) all correct, and therefore all equal, event (c_1, c_2),
ii) all wrong but all equal, event $(w_1, w_2$, equal),
iii) all wrong and unequal, event $(w_1, w_2$, unequal),

iv) one correct and one wrong, event (c_1, w_2) or (w_1, c_2).

The unison scheme accepts a pattern iff the two responses are the same. We obtain the following probabilities for the composite network:

$$p(\text{accept}) = p(c_1, c_2) + p(w_1, w2, \text{equal}), \quad (12)$$

$$p(\text{correct}|\text{accept}) = \frac{p(\text{correct, accept})}{p(\text{accept})} = \frac{p(c_1, c_2)}{p(c_1, c_2) + p(w_1, w_2, \text{equal})}. \quad (13)$$

In fact, events of the type (c_1, w_2) are always rejected because the response class cannot be equal. It is straightforward to generalize to the case of N networks, obtaining the following R-A values:

$$R = 1 - p(c_1, c_2, \ldots, c_N) - p(w_1, w_2, \ldots, w_N, \text{equal}) \quad (14)$$

$$A = \frac{p(c_1, c_2, \ldots, c_N)}{p(c_1, c_2, \ldots, c_N) + p(w_1, w_2, \ldots, w_N, \text{equal})} \approx 1 - \frac{p(w_1, w_2, \ldots, w_N, \text{equal})}{p(c_1, c_2, \ldots, c_N)} \quad (15)$$

where the last approximation is valid for a high signal-to-noise ratio, that is, for $p(w_1, w_2, \ldots, w_N, \text{equal}) \ll p(c_1, c_2, \ldots, c_N)$. For hypothesis ABA to be valid, the probability $p(w_1, w_2, \ldots, w_N, \text{equal})$ must be a small quantity, decreasing as the number of classifiers in the team grows.

In general, the performance of the unison scheme depends on the *joint* probability distribution for the output responses produced by input patterns extracted from the different classes. From an operational point of view, a specific scheme can be evaluated by combining the responses provided by the individual networks on a suitable test set and by calculating the probabilities needed in eqns (14) and (15). If the individual responses are available, the evaluation is trivial and requires a negligible effort, so that the actual test is suggested for a general recognition problem.

In Sections 4.2 and 4.6 we will investigate the possible performances that can be obtained after making some strong assumptions about the independence of the different nets and the distribution of mistakes. This permits estimates that depend only on the individual probabilities.

4.2. Independent Confusion

Let us start by assuming that the distribution of confused cases among the different classes is known. For each classifier n in a team of N classifiers let us define $p_n(\omega_j|\omega_i)$ as the conditional probability that a pattern is recognized as belonging to class ω_j given that the correct class is ω_i. The probabilities describe both the accuracy for the different classes and the spread of the wrong classifications among the incorrect classes. Probabilities of events with no indices refer to events of the composite classification system: $R \equiv p(\text{reject}) \equiv p(\text{reject}_{\text{team}})$.

We now quantify the results that can be obtained by the combined system in the approximation of independence among the different networks, so that the probability of a set of output responses given an input class is the product of the individual probabilities:

$$p(w_1, w_2, \ldots, w_N, \text{equal}) = \sum_{i=1}^{C} P(\omega_i) \sum_{j=1; j\neq i}^{C} \left[\prod_{n=1}^{N} p_n(\omega_j|\omega_i) \right], \quad (16)$$

$$p(w_1, w_2, \ldots, w_N) = \sum_{i=1}^{C} P(\omega_i) \prod_{n=1}^{N} p_n(\text{wrong}|\omega_i) = \sum_{i=1}^{C} P(\omega_i) \prod_{n=1}^{N} [1 - p_n(\omega_i|\omega_i)], \quad (17)$$

$$p(c_1, c_2, \ldots, c_N) = \sum_{i=1}^{C} P(\omega_i) \prod_{n=1}^{N} p_n(\omega_i|\omega_i). \quad (18)$$

From the above equations it is straightforward to derive estimates for the accuracy and rejection probabilities in a system composed of several classifiers.

4.3. Training and Test Sets

The task of OCR is both relevant for the applications and interesting as a benchmark of various classification techniques, including neural nets (see Sabourin & Mitiche, 1992, for recent results of omnifont type-written OCR and LeCun et al., 1989, for an application to handwritten zip code recognition). Our tests are in the domain of handwritten digit recognition. We describe the results of integrating up to five MLP neural nets, trained with on-line back propagation (Rumelhart, Hinton, & Williams, 1987), with different architectures and different input features extracted from the raw data, and a network trained with the learning vector quantization (LVQ) technique (Kohonen, 1990). The raw training set is composed of 6,496 images of handwritten digits (28×16 binary pixels), produced by several different writers, and the disjoint test set contains 12,981 images.

The architecture and individual performance on the test set of the six nets are as follows:

mlp1 28 input nodes, 28 hidden, 10 output. A = 94.71%.

mlp2 48 input nodes, 28 hidden, 10 output. A = 94.60%.

mlp3 32 input nodes, 64 hidden, 10 output. A = 93.17%.

mlp4 56 input nodes, 32 hidden, 10 output. A = 94.97%.

mlp5 45 input nodes, 45 hidden, 10 output. A = 94.83%.

lvq1 1000 codebook vectors, trained with an optimized LVQ algorithm.[3] A = 94.68%.

The features for **mlp1** and **lvq1** are obtained by dividing the original image into windows of dimension 4×4 and counting the number of black pixels in each window, normalized to obtain a value in the range [0, 1]. The features for **mlp5** are obtained in the same way, from a larger number of 4×4 windows, each having an overlap of one pixel with the neighboring ones. For **mlp4** the windows have dimension 4×2 and their total number is therefore doubled with respect to **mlp1**. **mlp2** is trained with multiscale features extracted from overlapped windows of varying size (two of dimension 16×16, 10 of 8×8 and 36 of 4×4). All the previous features were based on *gray pixels,* that is, numbers in [0, 1] proportional to the number of black pixels in a window. On the contrary, the input data for **mlp3** are obtained in a more complex way, using the crossing count and stroke proportion of different regions and peripheral features. These features have been adapted from those used by Zhang et al. (1989) for Chinese character recognition, to which the reader is referred for a detailed description.

Some examples of digits randomly extracted from the test set are shown in Figure 6, together with the features extracted for **mlp1**.

4.4. Testing the Unison Strategy—Independence Assumption

To simulate a situation with a high probability of mistakes caused by the presence of high noise levels in the input data, the original test patterns for nets **mlp1**, **mlp2**, and **mlp3** have been randomly substituted (with a 50% probability) with random patterns in the range [0, 1]. In this case, the individual accuracy rates for **mlp1**, **mlp2**, and **mlp3** are 52.17%, 51.91%, and 51.66%, respectively, so that rejecting the noisy patterns is crucial in order to provide a meaningful response.

In Table 1 and Table 2 we show both the observed R-A values together with the observed frequencies of the relevant events, and the values calculated from eqns (16–18) for the unison combination of the two nets **mlp1** & **mlp2** and of the nets **mlp1** & **mlp3**.

Note the significantly higher accuracy of the second combination (A = 87.71%) with respect to the first combination (A = 82.34%). This confirmed our expectation that combining **mlp1** with **mlp3**, which uses a complex and radically different set of features, should be more effective than combining it with **mlp2**, which uses features extracted with the same method, although from different windows.

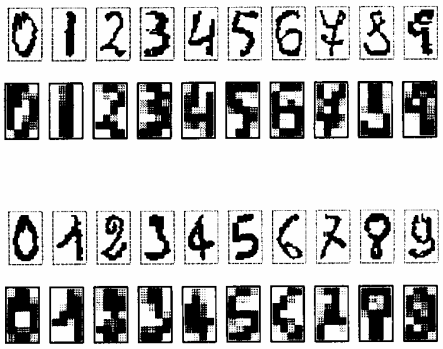


FIGURE 6. Examples of handwritten digits extracted from the test set: binary images and gray pixels features extracted from 4×4 windows (intensity proportional to value).

The method that bases the rejection on the output disagreement is illustrated in Figure 7, together with the pie-charts corresponding to the above results.

4.5. Testing the Unison Strategy—General Case

We tested various combinations of the nets described in Section 4.3. From the results, it is evident that a considerable boost in the system accuracy can be obtained from the integration of several classifiers (see Figure 8 for a summary of results).

In particular, the largest jumps in the accuracy tend to be obtained for the integration of networks using qualitatively different input features. For example, the integration of **mlp1** (gray pixels, trained with back propagation), **mlp3** (crossing counts, stroke proportion, and peripheral features, trained with back propagation) and **lvq1** (gray pixels, trained with the learning vector quantization algorithm) reaches the performance parameters (R = 11.75%, A = 99.48%). The global accuracy is increased by 4.77% with respect to the best of the three networks (**mlp1**) and by 6.31% with respect to the worst (**mlp3**). The incremental ratio with respect

TABLE 1
Combination of Two Nets (mlp1 & mlp2) to Reject Noisy Patterns

	Theoretical	Experimental
$p(c_1, c_2)$	27.38%	28.58%
$p(w_1, w_2, \text{equal})$	5.38%	6.13%
$p(w_1, w_2, \text{unequal})$	17.91%	18.37%
$p(c_1, w_2) + p(w_1, c_2)$	49.31%	46.90%
$p(\text{equal} \mid w_1, w_2)$	23.11%	25.02%
$p(\text{reject})$	67.60%	65.28%
$p(\text{correct} \mid \text{accept})$	83.59%	82.34%
$\Delta A / \Delta R$	0.46	0.46

[3] *LVQ_PAK: The Learning Vector Quantization Program Package.* Version 2.0 (January 31, 1991). Prepared by the LVQ Programming Team of the Helsinki University of Technology, Laboratory of Computer and Information Science, Rakentajanaukio 2 C, SF-02150 Espoo, Finland. Copyright (c) 1991.

TABLE 2
Combination of Two Nets (mlp1 & mlp3) to Reject Noisy Patterns

	Theoretical	Experimental
$p(c_1, c_2)$	27.02%	27.54%
$p(w_1, w_2, \text{equal})$	3.68%	3.85%
$p(w_1, w_2, \text{unequal})$	19.51%	19.85%
$p(c_1, w_2) + p(w_1, c_2)$	49.77%	48.74%
$p(\text{equal} \mid w_1, w_2)$	15.89%	16.27%
$p(\text{reject})$	69.38%	68.60%
$p(\text{correct} \mid \text{accept})$	88.00%	87.71%
$\Delta A/\Delta R$	0.51	0.51

to the initial situation of **mlp3** is $\Delta A/\Delta R = 0.53$ (see the arrows in Figure 8). Let us note that effective combinations do not require a large number of nets. In the test case, most of the advantage is gained by going from a single- to a double-net system, although a relatively small gain is obtained by adding a third net.

The performance characteristics in the previous tests are better than those that can be obtained from a single net by adding a threshold-based rejection mechanism. In Figure 9 we compare the range of possible $R - A$ points obtained by requiring that the maximum output value be higher than a specified threshold (see the thin line in Figure 9). Slightly better results are obtained by adding a second threshold on the difference between the maximum output and the second maximum (thres_diff = 0.2 in this case) (see the thick line in Figure 9).

The underlying hypothesis that the different networks tend to provide *different responses* when their classification is wrong is confirmed by the analysis of the distribution of the erroneous recognitions among the different classes. In Figure 10 we plot the observed relative frequencies with which three digits (0, 1, and 2) are confused with the other digits, for three different nets. For example, the digit 1 tends to be erroneously recognized as 7 by **mlp1** and **mlp3**, and it tends to be wrongly classified as 4 by net **mlp2** (the writing style of the data base is Italian, so that, with respect to the US style, digit 1 has an upper stroke and digit 7 a middle stroke cutting the vertical segment).

4.6. Uniform Confusion

In addition to the independence hypothesis of Section 4.2, let us assume that, in the case of an erroneous response, the output class is distributed with equal probability over the $(C - 1)$ possible wrong classes and

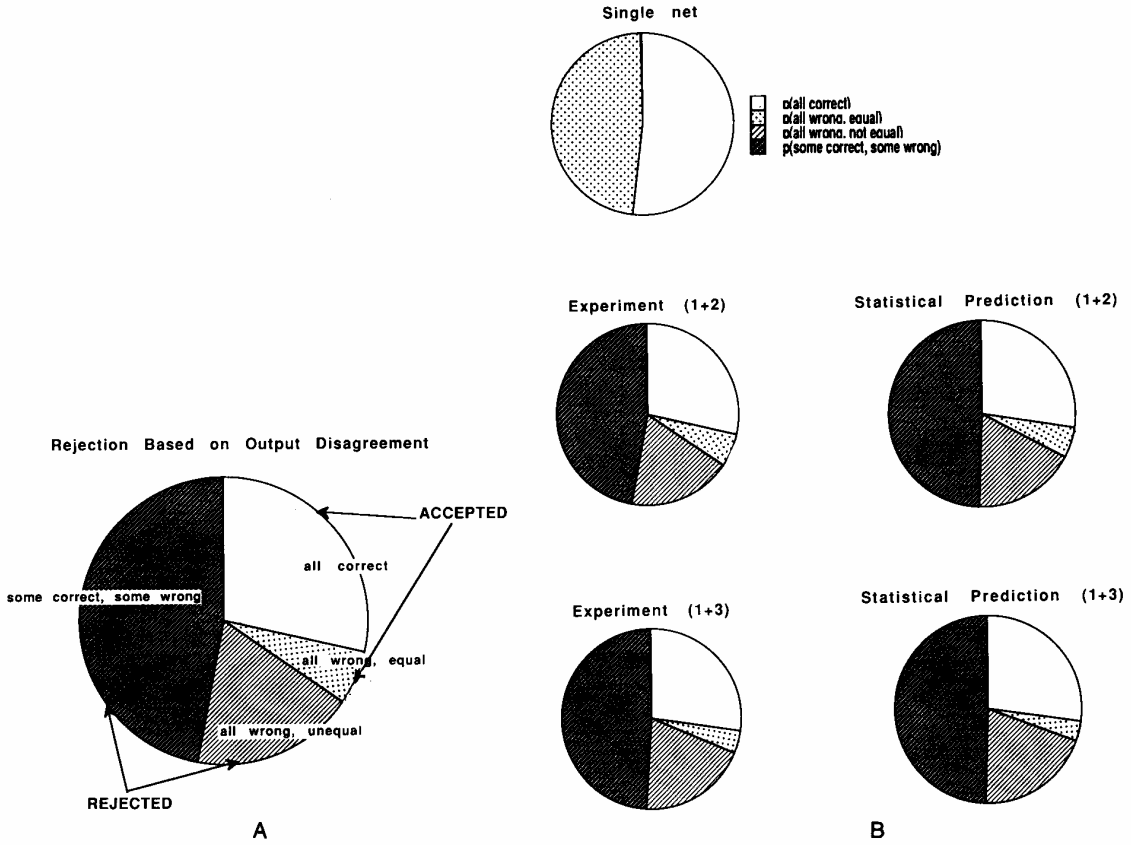


FIGURE 7. Illustration of (A) the disagreement-based rejection and (B) results for the combination of two nets on the noisy OCR problem.

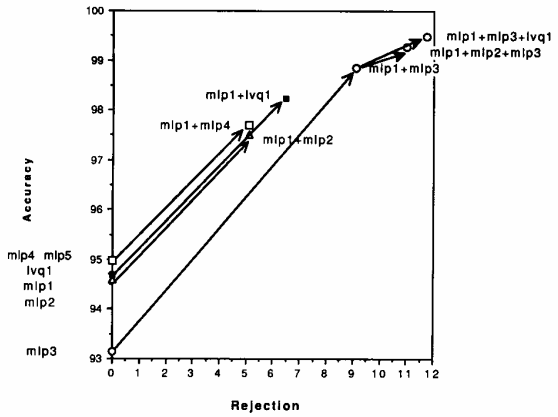


FIGURE 8. Unison rule for two and three nets. The accuracy gains (for larger rejection rates) are illustrated by arrows.

that this output is independent for the different networks. In this case, the probability of an equal response, given that all nets misclassify the pattern, tends to zero in the following way:

$$p(\text{equal}\,|\,w_1, w_2, \ldots, w_N) = \frac{1}{(C-1)^{(N-1)}}. \quad (19)$$

If the networks are independent with respect to the correctness of their response $[p(c_1, c_2, w_3, \ldots) = p(c_1)p(c_2)p(w_3)\ldots]$, from eqns (14) and (15) one obtains the following result:

$$1 - R \equiv p(\text{accept}) = \prod_{i=1}^{N} p(c_i) + \frac{\prod_{i=1}^{N} p(w_i)}{(C-1)^{(N-1)}}$$

$$= p(c_1)^N + \frac{p(w_1)^N}{(C-1)^{(N-1)}} \quad (20)$$

$$A \equiv p(\text{correct}\,|\,\text{accept}) = \frac{1}{1 + \dfrac{\prod_{i=1}^{N} p(w_i)}{(C-1)^{(N-1)} \prod_{i=1}^{N} p(c_i)}}$$

$$= \frac{1}{1 + \dfrac{p(w_1)^N}{(C-1)^{(N-1)} p(c_1)^N}} \quad (21)$$

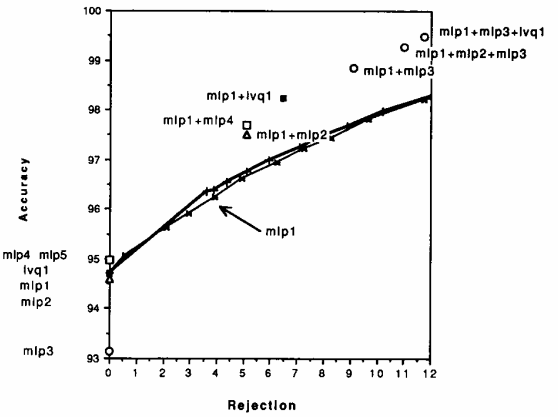


FIGURE 9. Comparison of unison combination with two-three nets versus threshold-based rejection for the mlp1 net.

where the last equality in eqns (20) and (21) is true if all networks have the same probability of correctness.

As an example, let us consider the combination of $N = 2$ nets with $p(c_i) = 0.9$ for a classification problem with $C = 10$ classes. In this case, the R-A values of the unison network are ($R = 0.188$, $A = 0.998$). In the different assumption that the networks are correlated so that $p(c_1, c_2) = p(c_1)$ and $p(w_1, w_2) = p(w_1)$, the values are ($R = 0.088$, $A = 0.987$). In this case, by starting from two sloppy networks with 90% accuracy we obtain a combined net with 98.7% accuracy, at the price of an 8.8% rejection ratio. Although the assumption of uniform confusion is very strong and difficult to realize in practical applications, in the last experimental part we will present some real-world examples for handwritten character recognition where the unison criterion is very effective even in the presence of weaker assumptions (see Sections 7 and 8).

5. MAJORITY RULES

Let us now relax the unison criterion to obtain a greater flexibility in the system design. Patterns are accepted if at least M classifiers agree on the classification. We furthermore assume that the majority is a strong one $[M > (N+1)/2]$, that the classifiers are independent, with equal performance $[c = p(\text{correct}_i),\ i = 1, \ldots, N,\ w = 1 - c]$, and that the output classes are maximally scattered in the case of a wrong classification (see the previous section).

In this case $p(\text{accept})$ is composed of a *signal* term, the probability that the number of correct answers is greater or equal to M, and a *noise* term, the probability that there exist at least a number M of equal *and* wrong classifications.

$$1 - R \equiv p(\text{accept})$$

$$= \sum_{i=M}^{N} \binom{N}{i} c^i w^{N-i} + \sum_{i=M}^{N} \binom{N}{i} w^i c^{N-i}$$

$$\times\, p(\text{equal responses} \geq M\,|\,w^i c^{N-i})$$

$$= \sum_{i=M}^{N} \binom{N}{i} c^i w^{N-i} + \sum_{i=M}^{N} \binom{N}{i} w^i c^{N-i}$$

$$\times \sum_{m=M}^{i} \binom{i}{m} \frac{1}{(C-1)^{m-1}} \left(\frac{C-2}{C-1}\right)^{i-m}. \quad (22)$$

The conditional probability of an accurate recognition $p(\text{correct}\,|\,\text{accept}) = p(\text{correct, accept})/p(\text{accept})$ can be derived by observing that $p(\text{correct, accept})$ is the probability that the acceptance occurs because of at least M equal *and* correct classifications, the signal term derived above.

$$A = \frac{p(\text{correct, accept})}{p(\text{accept})} = \frac{signal}{signal + noise}. \quad (23)$$

The above formulas can be approximated in many ways. If the number of classes is large, we can keep only a first-order approximation in $1/C$. Assuming that the

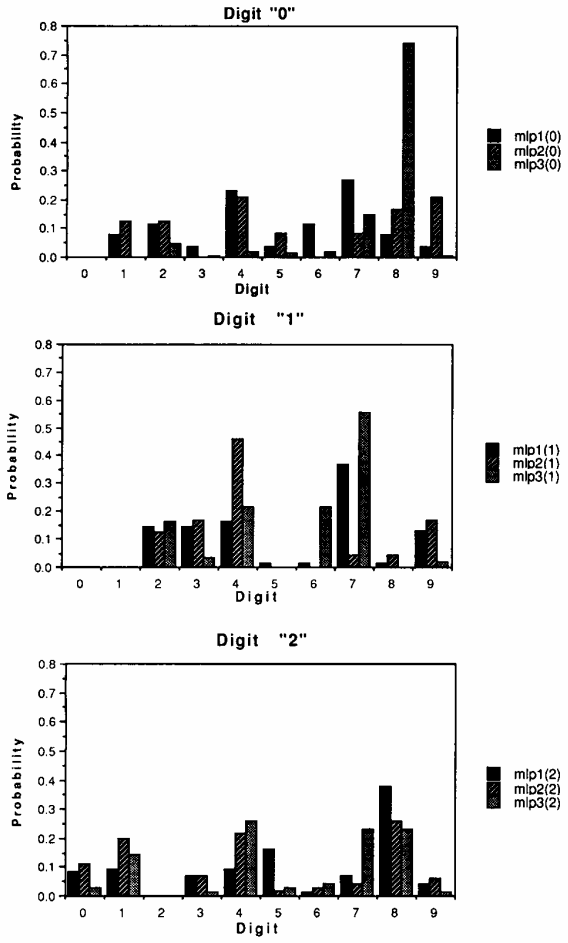


FIGURE 10. Uneven distribution of misclassifications among the possible wrong classes: $p(\omega_j|\omega_i)/[1-p(\omega_i|\omega_i)]$, for $j \neq i$. Observed frequencies are shown for the first three digits (0, 1, 2) using the MLP nets mlp1, mlp2, and mlp3. Note that the distributions are substantially diverse (especially for mlp1 and mlp3).

number C of classes tends to infinity, that M is equal to $N/2$ (N even), and that A is close to 1, we can approximate eqn (22) by keeping only the dominant term and using Stirling's approximation for the factorial:

$$1 - R \approx \binom{N}{M}(1-c)^{N-M}c^M \approx 2^N[c(1-c)]^{N/2}. \quad (24)$$

Additional approximations are described in Chernoff (1952). The above results need to be corrected if positive correlations are present in the different networks. In particular, if the true probability distributions $p(\text{pattern}|\text{class})$ are overlapping at the decision boundaries, the rejection rate in the limit of many nets needs to be different from zero to reach an accuracy close to 1.

5.1. A Test of Majority Rules

Here we present the experiment results obtained by applying the majority rule to the six nets described in Section 4.3. To increase the flexibility, two thresholds are varied during the tests. The first one refers to the minimum number of votes that must be collected by the winning class in order for the pattern to be accepted. The second threshold is a separation requirement that specifies a minimum difference between the votes of the winning class and the votes of the second one. All meaningful combinations of the two thresholds have been tested. In detail, the two thresholds for the seven tests plotted along the dashed line in Figure 11 are: (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), (5, 0), (6, 0). From the results displayed in Figure 11 it is evident that the performance can be better than that obtained from the unison combination, at the price of using a larger number of nets.

6. USE OF THE OUTPUT ACTIVATION VALUES

Up to this point, the only information that we used from the classifier was the response class, decided by the maximum output value. The previous schemes are therefore applicable to *any* classifier. Now, in some systems like MLP classifiers or statistical recognition systems, the output consists of continuous values in the range [0, 1] for the various classes, related to the posterior probability or confidence in the classification. In this section we show that, by using more of the output information provided, it is possible to build better final classifiers (with a similar recognition time).

The first observation is that the output values are related to the posterior probability for the different

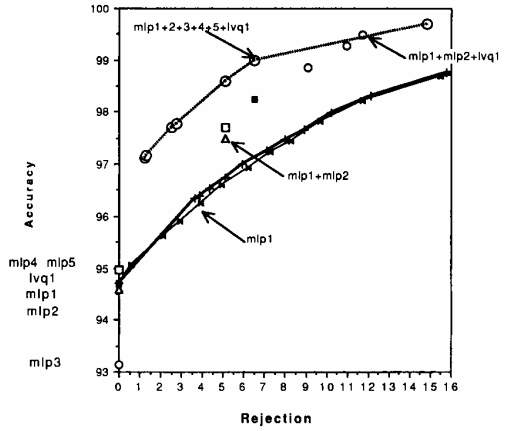


FIGURE 11. Performance of the majority rule with six nets (circles on dashed line). The results with the unison rule (two to three nets) and threshold-based rejection (one net) have been copied from Figure 9 for the comparison.

classes, given the input pattern. They are not *equal* to this probability both because of estimation errors (generalization errors) and because of some positive probability for the event *garbage input,* for example, occurring because of mistakes during the feature extraction phase. In our character recognition experiments, we observed that a preliminary normalization of the output values (so that they sum up to 1) leads to *worse* recognition results, confirming that some probability can be missing from the outputs. Methods for transforming the output levels to probability distributions are analyzed, for example, in Denker and leCun (1991).

Let's now remember that, in Bayes classification, the boundaries of the decision regions correspond to points where the maximum of $p(\text{class}|\text{input data})$ changes from one class to another. In addition, the recognition mistakes at a point in feature space are more frequent when the maximum over the different classes of $p(\text{class}|\text{input data})$ is low. If one manages to identify and reject the points that are near the boundaries of decision regions or the points where the probability density for the selected class is low, the final accuracy will be higher for the accepted patterns. The appropriate rejection rules derive from the above considerations. Considering a single network, let us define the acceptance rules **T1** and **T2** as follows:

- **T1** The maximum output value (*response*) must be higher than a fixed thresholds thres_max. The rule rejects data with low confidence for all classes.
- **T2** The difference between the activity levels of the two most active output units must be higher than a second threshold thres_diff. The rule rejects data that lead to an indecision between more classes.

Considering the combination of more networks, the patterns that survive criteria **T1** and **T2** for all networks in the team can be sieved by the unison criterion. In other words, a pattern is accepted only if all nets agree and all nets accept the pattern (i.e., all are certain in their conclusions). This scheme is called **T1&T2&U** (criteria **T1**, **T2** and unison).

A parallel version of rule **T1&T2&U** consists in replacing the individual checks (**T1**) with the check that the *average* of the maximum values of the different networks is larger than the threshold thres_max. Let's call **PT1** this parallel version of the rule and **PT1&T2&U** the global scheme.

Other possibilities are left for a further exploration. For example, one can substitute the unison with the majority requirements (**T1&T2&M**), or one can average the output vectors and apply criteria **T1** and **T2** to the averaged vector (**PT1&PT2**) (see Section 8).

7. MULTI-NET SYSTEMS BASED ON A UNIQUE SET OF FEATURES AND CLASSIFICATION ALGORITHM

Now we present an independent series of tests, both to increase the experimental basis for the combination approach and to test a more efficient scheme that uses the same features and the same learning and recognition algorithm for the different nets. With respect to the previous approaches, the use of a unique set of features and of a single training and recognition algorithm (MLP trained with *on-line* back propagation) allows a reduction in the preprocessing time and in the space (memory or hardware) required to realize the composite recognition algorithm. The only differences in the operation of the individual component networks are caused by the different net architectures (number of hidden units) and the different (random) initializations for training.

The rejection rules based on the confidence thresholds and on the agreement of different networks have been tested for the application of recognizing handwritten digits. Both the training and test set have been derived from a real application (automated document reading) and have realistic sizes (14,000 + 14,000 characters). The character images, digitized by an on-line scanner, are normalized to 30×22 binary pixels. Each image is represented by 70 gray pixels, where each gray pixel is derived by a 4×4 window that is overlapped with the neighboring ones for a one-pixel strip.

The MLP's architecture and performance is described by the following list.

- **net-1** $A = 98.21\%$. 70 input nodes, 100 hidden, 10 output.
- **net-2** $A = 98.08\%$. 70 input nodes, 100 hidden, 10 output.
- **net-3** $A = 97.24\%$. 70 input nodes, 70 hidden, 10 output.
- **net-4** $A = 97.98\%$. 70 input nodes, 100 hidden, 10 output.

First we present the results for the unison scheme based on the response class, then the results when the confidence in the classification (related to the analog output values) is also considered.

In Table 3 we show both the experimental R-A values, together with the observed frequencies of the relevant events, and the values calculated from eqns (16), (17), and (18) for the unison combination of **net-1** and **net-2**.

Although the ratio $\Delta A/\Delta R$ is close to the predicted one, it is apparent that the assumption of independence is not justified[4] and that the two networks tend to be

[4] In fact, $p(c_1, c_2)$ and $p(w_1, w_2)$ are much larger than the product of the two individual probabilities.

TABLE 3
Combination of Two Nets (net-1 & net-2) With the Same Architecture (70-100-10)

	Theoretical	Experimental
$p(c_1, c_2)$	96.32%	97.51%
$p(w_1, w_2$, equal)	0.007%	1.03%
$p(w_1, w_2$, unequal)	0.03%	0.18%
$p(w_1, c_2) + p(c_1, w_2)$	3.63%	1.27%
p(equal$\|w_1, w_2$)	19.07%	84.80%
p(reject)	3.67%	1.46%
p(correct|accept)	99.99%	98.95%
$\Delta A/\Delta R$	0.48	0.51

wrong in the same way [see the large experimental value for $p(\text{equal}|w_1 w_2)$]. This is not surprising and confirms the supposition that the behavior of the team classifier is almost independent of the initial random configuration (at least, in this particular learning experiment!) so that the two networks tend to produce the same mistakes. This is especially true if the mistakes are rare and caused by confusing writing styles (let us remember that the estimates are much closer to the experimental data in the high-noise situation described in Section 4.4). In this case, we suggest to use directly the estimates obtained by joining the results of different nets on the same test set.

In an effort to diversify the two networks, we substituted **net-2** with **net-3**, a net with a smaller number of hidden nodes. As it is shown in Table 4, the experimental $p(\text{equal}|w_1, w_2)$ is slightly reduced but remains large.

In spite of the strong correlations between the networks, the final *R-A* result is promising: in the first case the mistake rate on the accepted cases has been discounted by 41% (from 1.79%, using **net-1** only, to 1.05%) at the price of a rejection rate of 1.46%!

Increasing the number of networks does not produce very different results. Using **net-1**, **net-2**, **net-3** the accuracy reaches 99.13%; adding **net-4** it reaches 99.26% ($R = 2.99\%$).

Let us now consider the threshold-based rejection schemes that take into account the vector of output values for the different classes. First, considering the scheme **T1&T2** applied to classifier **net-1**, we obtain the *R-A* results shown in Figure 12, where we show two curves obtained with different values (0.0 and 0.2) of the thres_diff parameter. The rejection increases by increasing the thres_max value.

TABLE 4
Combination of Two Nets (net-1 & net-3) With a Different Number of Hidden Units

	Theoretical	Experimental
$p(c_1, c_2)$	95.48%	96.80%
$p(w_1, w_2$, equal)	0.01%	1.10%
$p(w_1, w_2$, unequal)	0.04%	0.27%
$p(w_1, c_2) + p(c_1, w_2)$	4.46%	1.82%
p(equal$\|w_1, w_2$)	20.42%	79.79%
p(reject)	4.52%	2.1%
p(correct|accept)	99.99%	98.88%
$\Delta A/\Delta R$	0.39	0.32

For example, a performance of $A = 99.47\%$ is reached with parameters thres_max = 0.9, thres_diff = 0.2. The corresponding rejection is 4.53%.

Using the scheme **PT1&T2&U** with a combination of **net-1** and **net-2** we obtain strictly better results (Figure 13) for different values of thres_diff.

To make a comparison with the single-network case, a performance of $A = 99.47\%$ is reached with parameters thres_max = 0.7, thres_diff = 0.2. The corresponding rejection is 3.50%. The same performance is obtained by rejecting 1% less cases. For this problem, the performance does not ameliorate with the addition of another classifier, in fact it decreases, also because the individual accuracies of **net-3** and **net-4** are less than those of the two other networks.

The advantage of the double-net system over the single-net one is shown in Figure 14, where we compare the two cases (with thres_diff = 0.2).

By using the analog output values in addition to the unison criterion, the error rate was discounted by 72% (from 1.79% to 0.53%) with a small rejection rate ($R = 3.50\%$). The $\Delta A/\Delta R$ value is 0.36, comparable with the ones obtained by using only the unison criterion, although the accuracy is now substantially higher.

8. AVERAGING NETWORK OUTPUTS

Another way to combine several networks is by *averaging* their responses.[5] The procedure is straightforward: the output vectors $\vec{o}^{(i)} = (o_1^{(i)}, o_2^{(i)}, \ldots, o_M^{(i)})$, $i = 1, \ldots, N$ of the networks are averaged, producing an **average output** vector $\vec{\tilde{o}} = (\tilde{o}_1, \tilde{o}_2, \ldots, \tilde{o}_M)$. Classification will be accomplished by assigning the unknown pattern to class $\tilde{j}$ if, for each i, $\tilde{o}_i \leq \tilde{o}_{\tilde{j}}$.

Criteria **T1** and/or **T2** can be applied to the averaged vector $\vec{\tilde{o}}$ (scheme **PT1&PT2**). Let us note that rule **PT1**, that operates with averaged networks, results in a *stronger* (i.e., more restrictive) condition than the one of rule **T1** with a team of single nets. All patterns accepted with **PT1** are accepted with **T1**, but not *vice versa*. In fact, if **PT1** accepts one pattern, the *average* output for the winning class is greater than the threshold values and therefore at least one value of a particular net is greater than T_{accept} and the pattern is accepted by at least one network.

When the unison criterion is used, if a pattern is accepted by *all* nets with the same classification, the

[5] Averaging also smoothes out the effects of random network initialization, thus increasing the system reliability.

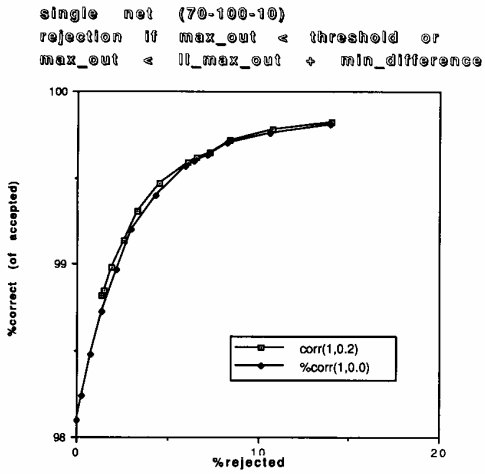


FIGURE 12. Single MLP net: comparison between different values for thres_diff. The rejection rate increases with growing thres_max values.

average output for the winning class is trivially greater than T_{accept}, so that rule **T1&U** behaves exactly in the same way as rule **PT1&U**. On the other hand, rule **T2&U** is *weaker* (i.e., accepts more patterns) when operating with averaged networks (**PT2&U**). In fact, if all nets satisfy **T2**, for the net n and the winning class w one has $o_w(n) > o_i(n)$ + thres_diff, for $i \neq w$. After averaging over all nets one obtains: $\tilde{o}_w > \tilde{o}_i$ + thres_diff, so that criterion **PT2** is satisfied.

If the two rules are combined, the scheme (**PT1&PT2**) can reduce R with respect to rule (**PT1&T2**), but it can also decrease the accuracy A. This has been confirmed by further experimental results (Table 5), obtained with two large nets, trained on a set of 28,000 characters, doubling the size that was used

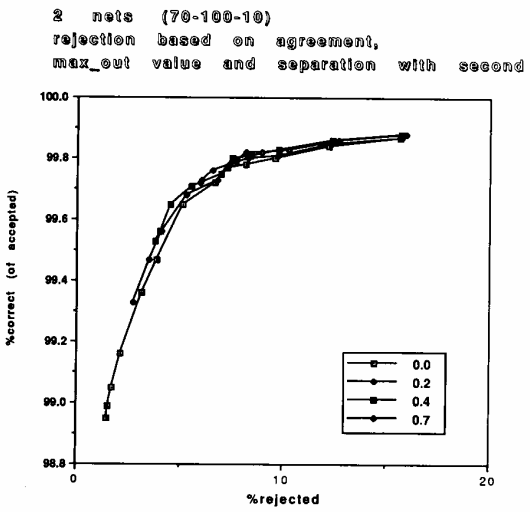


FIGURE 13. Two MLP nets: comparison between different values for thres_diff.

in the experiments described in the previous section, and tested on another large, disjoint test set (49,000 characters). These data sets are part of the same handwritten digit data base and have been preprocessed the same way as in the tests described in Section 7. The two individual nets yield an individual recognition performance of 99.06% and 99.01%, respectively. It is evident that the application of thres_max with thresh_diff = 0 has the same effect with both schemes (apart from the effects of finite-precision arithmetic). On the other hand **PT2** is less restrictive than **P2** (compare R and A with team and average when thres_diff is greater than 0). The choice of the most appropriate scheme depends on the specific application requirements.

9. CONCLUSIONS AND RELATIONS WITH PREVIOUS WORK

Decisions taken by teams usually are better than decisions taken by individuals, provided that suitable methods for combining the individual responses are provided. In this paper we considered different ways to combine the output of different neural classifiers both to increase the flexibility in the rejection–accuracy rates for a selected application and to obtain a combined performance that is higher than that obtainable from the individual components.

Part of the motivation for combining the outputs of different networks can be derived from the usual statistical technique of lowering the variance of estimates by averaging. For example, neural network portfolios are suggested in Mani (1991). But the more interesting motivation is related to the *integration* of different recognition techniques (based on different sets of features, architectures, learning algorithms, etc.) to reach performances that are higher than the best obtainable by the individual nets. Although some suggestions about the potentially fruitful combinations can be obtained from probabilistic estimates and assumptions about the uncorrelation of the individual mistakes, the combined performance depends on the joint probability distribution of the outputs. But estimating the joint probability requires the test of different nets on the same test set, so that one may as well estimate directly the combined performance. This operation is very fast provided that the detailed results for all patterns in a test set are saved for the different nets.

The presented approach is different and less ambitious from that advocated by Wolpert (1992), where the outputs of more nets are used as input patterns for training a higher-level generalizer, and from that presented by Jacobs and Jordan (1991) and Jacobs et al. (1991), where the different expert networks and the gating network are trained together with the use of competition (thereby favoring specialization). Ensemble networks, that is, replicated networks in which a

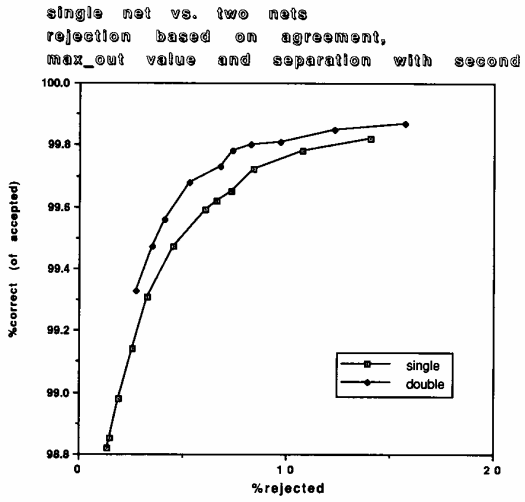


FIGURE 14. Comparison between single- and double-net system.

number of identical nets are independently trained on the same data and their results averaged, are considered in Pearlmutter and Rosenfeld (1991). In this case, the averaging operation on the outputs is considered as a way to reduce the Chaitin–Kolmogorov complexity of the resulting classifier, therefore increasing the expected generalization. In the case of back propagation networks, the complexity of the classifier exceeds that of the training data plus the complexity of the learning procedure because of the introduction of randomly broken symmetries, for instance, the random initialization. An application of majority nets for breaking cryptosystems is presented in Apolloni, Cesa-Bianchi, and Ronchini (1990) and some preliminary results in OCR are contained in Battiti and Colla (1992). Ensembles of large numbers (up to about 15) of look-up table networks with the consensus rule are considered in Hansen, Liisberg, and Salamon (1992) for handwritten digit recognition. The framework of our experiments is different and the ensemble approach is not appropriate in our case because we consider only small sets with networks of different kinds, whose choice is ultimately heuristic, although guided by statistical assumptions.

The main lesson that we learned from an extensive experimentation based on the environment described in Battiti et al. (1991) is that the integration of a small number of different *independently trained* modules often is a fast and effective way to reach the rejection–accuracy rates required by an application by starting from a data base of neural networks (and possibly standard recognizers!) developed for the same task. This portfolio of nets is a byproduct of a typical development phase, where different features and/or architectures are tested. Last, but not least, some of the proposed methods permit a straightforward cooperation of traditional and neural classifiers (that can be expected to have a high degree of uncorrelation) and can facilitate the adoption of neural nets in application environments.

TABLE 5
Combination of Two Nets: Team (PT1 & T2) vs. Averaging (PT1 & PT2)

	thres_max	thres_diff	*R*	*A*
2-Net team	—	—	0.69	99.45
	0.9	—	2.67	99.83
	—	0.2	1.41	99.68
	0.9	0.2	2.79	99.85
Average	—	—	0.0	99.11
	0.9	—	2.64	99.81
	—	0.2	0.69	99.47
	0.9	0.2	2.69	99.83

REFERENCES

Apolloni, B., Cesa-Bianchi, N., & Ronchini, G. (1990). Training neural networks to break the knapsack cryptosystem. In E. R. Caianiello (Ed.), *Proceedings of the III Italian Workshop on Parallel Architectures and Neural Networks* (pp. 377–382). Singapore: World Scientific.

Battiti, R., Briano, L. M., Cecinati, R., Colla, A. M., & Guido, P. (1991). An application-oriented development environment for neural net models on the multiprocessor Emma-2. In M. Sami, & J. Calzadilla-Daguerre (Eds.), *Silicon Architectures for Neural Nets—Proceedings IFIP WG* 10.5 *Workshop, Saint Paul de Vence, France, November,* 1990 (pp. 31–43). Amsterdam: North-Holland.

Battiti, R., & Colla, A. M. (1992). On the cooperation of multiple nets for pattern recognition. In E. R. Caianiello (Ed.) *Proceedings of the V Workshop Reti Neurali WIRN Vietri-92, Vietri sul Mare, Salerno, Italy, May* 1992 (pp. 237–243). Singapore: World Scientific.

Burrascano, P. (1991). Learning vector quantization for the probabilistic neural network. *IEEE Transactions on Neural Networks,* **2**(4), 458–461.

Chernoff, H. (1952). A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics,* **23,** 493–507.

Denker, J. S., & leCun, Y. (1991). Transforming neural-net output levels to probability distributions. In R. P. Lippmann, J. Moody, & D. S. Touretzky (Eds.), *Advances in neural information processing systems—NIPS* 3 (pp. 853–859). San Mateo, CA: Morgan Kaufmann.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis.* New York: John Wiley & Sons.

Hansen, L. K., Liisberg, C., & Salamon, P. (1992). Ensemble methods for handwritten digit recognition. In S. Y. Kung, F. Fallside, J. A. Sorenson, & C. A. Kamm (eds.) *Neural Networks for Signal Processing II, Proceedings of the* 1992 *IEEE-SP Workshop* (pp. 333–342). Piscataway, NJ: IEEE Service Center.

Jacobs, R. A., & Jordan, M. I. (1991). A competitive modular connectionist architecture. In R. P. Lippmann, J. Moody, and D. S. Touretzky (Eds.), *Advances in neural information processing systems—NIPS* 3 (pp. 767–773). San Mateo, CA: Morgan Kaufmann.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation,* **3**(1), 79–87.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE,* **78**(9), 1464–1480.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation,* **1**(4), 541–551.

Mani, G. (1991). Lowering variance of decisions by using artificial network portfolios. *Neural Computation,* **3**(4), 484–486.

Pearlmutter, B. A., & Rosenfeld, R. (1991). Chaitin–Kolmogorov complexity and generalization in neural networks. In R. P. Lippmann, J. Moody, & D. S. Touretzky (Eds.), *Advances in neural information processing systems—NIPS* 3 (pp. 925–931). San Mateo, CA: Morgan Kaufmann.

Ruck, D. W., Rogers, S. K., Kabrisky, M., Oxley, M. E., & Suter, B. W. (1990). The multilayer perceptron as an approximation to a Bayes optimal discriminant function. *IEEE Transactions on Neural Networks,* **1**(4), 296–298.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1987). Learning internal representations by error propagation. In D. E. Rumelhart, & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (pp. 318–362). Cambridge, MA: MIT Press.

Sabourin, M., & Mitiche, A. (1992). Optical character recognition by a neural Network. *Neural Networks,* **5**, 843–852.

Wan, E. A. (1990). Neural network classification: A Bayesian interpretation. *IEEE Transactions on Neural Networks,* **1**(4), 303–304.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks,* **5**, 241–259.

Zhang, Z., Hartmann, I., Guo, J., & Suchenwirth, R. (1989). A recognition method of printed chinese characters by feature combination. *International Journal of Research and Engineering—Postal Applications,* **1**, 77–82.

APPENDIX A

Here we sketch the demonstration of the optimality of the threshold-based rejection of eqn (3).

Let us assume that we have C decision regions for the different classes and that the current rejected region $\mathcal{R}$ corresponds to an accuracy $A = p(\text{correct, accept})$. If the rejected region is enlarged by a small portion $\Delta\mathcal{R}$, one obtains (in a first-order approximation):

$$\Delta p(\text{correct}\,|\,\text{accept}) = \Delta\frac{p(\text{correct, accept})}{p(\text{accept})} \approx -\frac{p(\text{correct, accept})}{p(\text{accept})^2}\Delta p(\text{accept}) + \frac{\Delta p(\text{correct, accept})}{p(\text{accept})} \quad \text{(A.1)}$$

After substituting the following integrals for $\Delta p(\text{reject})$ and $\Delta p(\text{correct, accept})$:

$$\Delta p(\text{correct}\,|\,\text{accept}) = -\int_{\Delta\mathcal{R}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x} \quad \text{(A.2)}$$

$$\Delta p(\text{reject}) = \int_{\Delta\mathcal{R}} p(\mathbf{x})\,d\mathbf{x} \quad \text{(A.3)}$$

and remembering that $\Delta p(\text{accept}) = -\Delta p(\text{reject})$ one obtains:

$$\frac{\Delta p(\text{correct}\,|\,\text{accept})}{\Delta p(\text{reject})} \approx \frac{1}{p(\text{accept})}\left[p(\text{correct}\,|\,\text{accept}) - \frac{\int_{\Delta\mathcal{R}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x}}{\int_{\Delta\mathcal{R}} p(\mathbf{x})\,d\mathbf{x}}\right] \quad \text{(A.4)}$$

$$\approx \frac{1}{p(\text{accept})}\left[\frac{\int_{\mathcal{A}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x}}{\int_{\mathcal{A}} p(\mathbf{x})\,d\mathbf{x}} - \frac{\int_{\Delta\mathcal{R}} \max_i p(\omega_i|\mathbf{x})p(\mathbf{x})\,d\mathbf{x}}{\int_{\Delta\mathcal{R}} p(\mathbf{x})\,d\mathbf{x}}\right] \quad \text{(A.5)}$$

where $\mathcal{A}$ is the accepted region. In other words, the ratio is proportional to the average of $\max_i p(\omega_i|\mathbf{x})$ over the accepted region minus the average over the additional rejected region. In the approximation that the probability densities are constant over a small region centered on $\mathbf{x}^*$, one obtains:

$$\frac{\Delta p(\text{correct}\,|\,\text{accept})}{\Delta p(\text{reject})} \approx \frac{1}{p(\text{accept})}\left[p(\text{correct}\,|\,\text{accept}) - \max_i p(\omega_i|\mathbf{x}^*)\right]. \quad \text{(A.6)}$$

The ratio is maximized when the additional rejection region $\Delta\mathcal{R}$ has the lowest probability for the winning class, that is, the lowest value of $\max_i p(\omega_i|\mathbf{x})$. Therefore, increasing portions of the input space are rejected in an optimal way by starting from the patterns with the lowest $\max_i p(\omega_i|\mathbf{x})$ values and then including patterns with larger and larger values.

This means that a given rejection rate gives the best accuracy if the rejected patterns are those that satisfy:

$$\max_i p(\omega_i|\mathbf{x}) < T_{\text{accept}} \quad \text{(A.7)}$$

for a given threshold value T_{accept}.

A growing fraction is rejected when the threshold increases. The criterion is intuitive because, after increasing the rejected region, the number of additional rejected cases is proportional to $p(\mathbf{x})$ and, among these, the number that would have been classified in the wrong way is proportional to $[1 - \max_i p(\omega_i|\mathbf{x})]$. One must reject first the cases that, with a high probability, would cause recognition mistakes.

Let us consider what happens when increasing portions of the space are rejected according to the optimal threshold-based criterion (see Figure A.1 for a graphical illustration).

If one has C possible classes, the probability for the winning class must be larger than or equal to $1/C$ [trivial from the definition of winning class as the class with the largest $p(\omega_i|\mathbf{x})$]. Therefore, if the threshold T_{accept} is less than $1/C$, no patterns will be rejected. Let us denote as T_{start} the largest threshold for which no patterns are rejected [$T_{\text{start}} = \inf_x \max_i p(\omega_i|\mathbf{x})$]. In this case, starting from eqn (A.6) and assuming that the derivative exists, one derives:

$$\left.\frac{dp(\text{correct}\,|\,\text{accept})}{dp(\text{reject})}\right|_{R=0} = [p(\text{correct}\,|\,\text{accept}) - T_{\text{start}}] = [A_0 - T_{\text{start}}] \quad \text{(A.8)}$$

where A_0 is the accuracy when all patterns are accepted. The initial derivative of the function $A(R)$ is equal to the accuracy over all patterns [the global average of $\max_i p(\omega_i|\mathbf{x})$] minus the initial threshold (equal

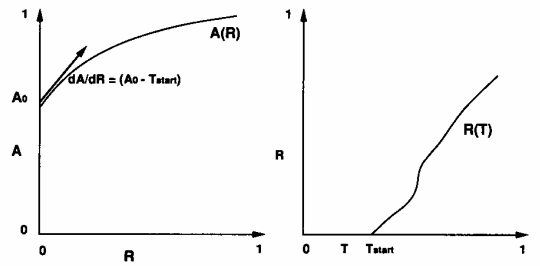


FIGURE A1. Relations between the accuracy–rejection curve $A(R)$ and the rejection–threshold curve $R(T)$.

to $\inf_x \max_i (\omega_i | \mathbf{x})]$. Remembering the definitions and the inequality on T_{start} one derives:

$$\left.\frac{dA}{dR}\right|_{R=0} \leq \left[A_0 - \frac{1}{C}\right] \tag{A.9}$$

For a large number of classes the derivative *can* be close to 1 for large values of A_0. *Vice versa*, for two classes it cannot be larger than $\frac{1}{2}$, at least one good classification must be lost for every avoided misclassification.

When the rejected portion increases:

$$\frac{dA}{dR} = \frac{1}{(1-R)}[A - T(R)] \tag{A.10}$$

where $T(R)$ is the threshold value for a fraction R of rejected patterns. $T(R)$ can be estimated by inverting the $R(T)$ function, which gives the rejection rate for increasing values of the threshold.

From eqn (A.5) it is easy to check that the derivative is nonnegative if the (increasing) threshold is used for accepting patterns. In fact, all points in the rejected region have $\max_i p(\omega_i | \mathbf{x}) < T_{\text{accept}}$, and all points in the accepted region have $\max_i p(\omega_i | \mathbf{x}) \geq T_{\text{accept}}$. Therefore, the average of $\max_i p(\omega_i | \mathbf{x})$ over the accepted region $\mathcal{A}$ cannot be less than the average over the additional rejected region $\Delta\mathcal{R}$.

NOMENCLATURE

Probability and Classification Task

C	number of classification classes
ω_i	classification class ($i = 1, \ldots, C$)
$\mathcal{R}_i$	ith decision region
$P(\omega_i)$	a priori probability for the ith class
$p(\mathbf{x} \mid \omega_i)$	*state-conditional* probability density
$P(\omega_i \mid \mathbf{x})$	*posterior* probability
$p(\mathbf{x})$	global probability density
$P(\text{correct})$	global probability of a correct recognition
$A = p(\text{correct} \mid \text{accept})$	probability of an accurate response, given that the input pattern is accepted
$R = p(\text{reject})$	probability that a pattern is rejected by the classifier
$\Delta A, \Delta R$	differences of A, R corresponding to different classifiers
$\mathcal{U} \stackrel{\text{def}}{=} A - \lambda R$	performance function
$\mathcal{R}$	rejection region
$\mathcal{A}$	acceptance region
P_i	probability of classifier i in a combination of more classifiers
$\delta(P_1)$	difference between the accuracy of the combined classifier and the linear combination of accuracies
T_{accept}	threshold for accepting a pattern
N	number of classifiers
M	number of classifiers that must have the same response
$p(w_1, w_2, \ldots, w_N, \text{equal})$	probability that N classifiers have an equal response that is wrong
$p(w_1, w_2, \ldots, w_N)$	probability that N classifiers have wrong responses
$p(c_1, c_2, \ldots, c_N)$	probability that N classifiers have correct responses

Normal Distribution

d	dimension of the space
$\mathbf{m} = E[\mathbf{x}]$	mean vector
$\Sigma = E[(\mathbf{x} - \mathbf{m}) \times (\mathbf{x} - \mathbf{m})^t]$	covariance matrix of the multivariate normal density
μ	mean vector of one-dimensional normal distribution
σ^2	variance of one-dimensional normal distribution
$N(\theta, \mu, \sigma)$	one-dimensional normal distribution

Functions

$\theta(x)$ function equal to 1 if $x \geq 0$, 0 otherwise