V. Petridis and Ath. Kehagias.
"A Recurrent Network Implementation of Bayesian Time Series Classification".

This paper has appeared in the journal:
Neural Computation, vol.8, pp.357-372, 1996.

A Recurrent Network Implementation of Time Series Classification

Vas. Petridis
Div. of Electronics and Comp. Eng., Dept. of Electrical Eng.
Aristotle University of Thessaloniki
540 06 Thessaloniki, Greece
and
Ath. Kehagias
Div. of Electronics and Comp. Eng., Dept. of Electrical Eng.
Aristotle University of Thessaloniki
540 06 Thessaloniki, Greece
and
Dept. of Mathematics
American College of Higher Studies
540 06 Pylea, Greece
e-mail: kehagias@egnatia.ee.auth.gr

15 June 1996

Abstract

An Incremental Credit Assignment (ICRA) scheme is introduced and applied to time series classification. It has been inspired from Bayes' rule, but the Bayesian connection is not necessary either for its development or proof of its convergence properties. The ICRA scheme is implemented by a recurrent, hierarchical, modular neural network, which consists of a bank of predictive modules at the lower level, and a decision module at the higher level. For each predictive module, a credit function is computed; the module which best predicts the

observed time series behavior receives highest credit. We prove that the credit functions converge (with probability one) to correct values. Simulation results are also presented.

amstex

This paper appears in *Neural Computation*, vol.8, pp.357-372, 1996.

1 Introduction

Consider the following problem of time series classification. A time series y_t, $t = 1, 2, ...$ is produced by a source $S(\theta_k)$, where θ_k is a parameter taking values in a *finite* set $\Theta = \{\theta_1, ..., \theta_K\}$ and the "true" or "best" value of θ_k is sought. This problem appears in many practical applications, e.g. speech recognition (Rabiner & Schafer, 1988), enzyme classification (Papanicolaou & Medeiros, 1990) etc. An extensive list of applications can be found in (Hertz, Krogh & Palmer, 1991). In this paper we present an *Incremental CRedit Assignment* (ICRA) scheme which assigns credit to each source according to its predictive power. This approach yields a hierarchical architecture with a prediction level at the bottom and a decision level at the top. We present a recurrent, hierarchical, modular neural network implementation of this architecture. A bank of local prediction modules are trained, each on data from a particular source $S(\theta_k)$. The prediction modules can be implemented by several different kinds of feedforward neural networks: sigmoid, linear, Gaussian, polynomial etc. The decision module is implemented by a recurrent Gaussian network which combines the outputs of the prediction modules. The overall structure of the network is presented in Fig. 1. We prove that the credit functions converge with probability one to correct values, namely, to one for the module with maximum predictive power and to zero for the remaining modules. Moreover ICRA has an easy neural network implementation (using only adders and multipliers). ICRA has been inspired by classification based on the Bayesian posterior probabilities of the candidate sources, but the Bayesian connection is not necessary for developing ICRA or for proving its convergence properties.

The idea of combining local models into a large modular network has recently become very popular. It is used for prediction as well as for classification of both static and dynamic (time series) patterns. Early examples of this idea are, for example, (Farmer & Sidorowich, 1988; Moody, 1989) where a time series *prediction* problem is solved by partitioning the *input* space into a number of regions and training a local predictor for each region; in every instance, the local predictor used is explicitly determined by past input values, hence it is not necessary to assign credit to each predictor. A later development is the combination of local models using a weighted sum; the weights can be interpreted as conditional probabilities or as credit functions. This is the approach taken in (Jacobs et al., 1991; Jordan & Jacobs, 1992; Neal, 1991; Nowlan, 1990) where the terms *local experts* and *probability mixtures* are used; the term *committees* appears in (Schwarze & Hertz, 1992); the term *neural ensembles* in (Perrone & Cooper, 1993), and so on. Our point of view is similar to that of the above papers, insofar we also use local models (predictors) and credit functions. However, ICRA is a *recursive* scheme for *online* credit assignment, so that classification at a given time depends on past classifications. This is particularly appropriate for classification of *dynamic* patterns, such as time series, where the history of the signal must be taken into account. In contrast, the abovementioned papers use offline credit assignment and are applied either to static problems or "staticized" dynamic problems, where preprocessing is used to transform a time-evolving signal to a static feature vector (FFT or LPC coefficients etc.). However, static feature vectors may not capture all the dynamic properties of a time series, especially in case of source switchings. On the other hand, while our method assumes that the classes to be used are given in advance, several of the abovementioned papers present algorithms that *discover* an expedient partition of the source space. In fact, there are several neural algorithms which combine local models and *adaptive* partitioning; for example (Ayestaran & Prager, 1993; Baxt, 1992; Jordan & Jacobs, 1994; Kadirkamanathan & Niranjan, 1992; Schwarze & Hertz, 1992; Shadafan & Niranjan, 1993) etc. However, while such algorithms perform adaptive *par-*

titioning, they do not perform, as far as we know, adaptive *classification*, since they do not use classification results recursively.

In short, our ICRA algorithm is applicable to problems of time series classification, where past classification results must be used for future classification, and classes are given in advance.

2 Bayesian Time Series Classification

A random variable Z which takes values in $\Theta = \{\theta_1, ..., \theta_K\}$ is introduced. The time series y_1, y_2, ... is produced by source $S(Z)$. For instance, if $Z = \theta_1$, then the time series y_1, y_2, ... is produced by $S(\theta_1)$. At every time t a decision rule produces an estimate of Z, denoted $\hat{Z}_t$. For instance, if *at time t* we believe that the time series y_1, ... y_t has been produced by θ_1, then $\hat{Z}_t = \theta_1$. Clearly, $\hat{Z}_t$ may change with time, as more observations become available.

The *conditional posterior probability* p_t^k for $k = 1, 2, ..., K$, $t = 1, 2, ...$ is defined by

$$p_t^k \doteq Prob(Z = \theta_k \mid y_t, ..., y_1);$$

also the *prior probability* p_0^k for $k = 1, 2, ... , K$ is defined by

$$p_0^k \doteq Prob(Z = \theta_k \mid at \;\; t = 0). \tag{1}$$

p_0^k reflects our prior knowledge of the value of Z. In the absence of any prior information we can just assume all models to be equiprobable: $p_0^k = 1/K$ for $k = 1, 2, ..., K$. p_t^k reflects our belief (after observing data $y_1, ..., y_t$) that the time series is produced by $S(\theta_k)$. We choose $\hat{Z}_t \doteq \arg\max_{\theta_k \in \Theta} p_t^k$. In other words, at time t we consider that $y_1, ..., y_t$ has been produced by source $S(\hat{Z}_t)$, where $\hat{Z}_t$ maximizes the posterior probability. So the classification problem has been reduced to computing p_t^k, $t = 1, 2, ..., k = 1, 2, ..., K$. This computation (Hilborn & Lainiotis, 1969; Lainiotis & Plataniotis, 1994)

is based on Bayes' rule:

$$p_t^k = \frac{Prob(y_t, Z = \theta_k \mid y_{t-1}, ..., y_1)}{\sum_{j=1}^{K} Prob(y_t, Z = \theta_j \mid y_{t-1}, ..., y_1)}. \tag{2}$$

Also

$$Prob(y_t, Z = \theta_k \mid y_{t-1}, ..., y_1) = Prob(y_t \mid y_{t-1}...y_1, Z = \theta_k) \cdot p_{t-1}^k. \tag{3}$$

Now (2), (3) imply the following recursion for k =1,2, ..., K, t =0, 1, 2, ... :

$$p_t^k = \frac{Prob(y_t \mid y_{t-1}, ..., y_1, Z = \theta_k) \cdot p_{t-1}^k}{\sum_{j=1}^{K} Prob(y_t \mid y_{t-1}, ..., y_1, Z = \theta_j) \cdot p_{t-1}^j}. \tag{4}$$

and we only need (for each t and k) to compute $Prob(y_t \mid y_{t-1}, ..., y_1, Z = \theta_k)$. This probability depends on the form of the predictor; the predictors have a general parametric form $f(\cdot; \theta_k)$, $k = 1, ... , K$:

$$y_t^k = f(y_{t-1}, ..., y_{t-N}; \theta_k). \tag{5}$$

Typically, $f(\cdot; \theta_k)$ would be a feedforward (linear, sigmoid, Gaussian, polynomial) neural network trained on data from source $S(\theta_k)$. This predictor approximates y_t *when the time series is produced by* $S(\theta_k)$. For $k = 1, 2, ..., K$ the prediction error e_t^k, $k = 1, ... , K$, $t = 1, 2, ...$ is defined by

$$e_t^k \doteq y_t - y_t^k. \tag{6}$$

It is *assumed* that e_t^k is a white, Gaussian noise process, with conditional probability of the form

$$Prob(e_t^k \mid y_{t-1}, ..., y_1; Z = \theta_k) = C(\sigma_k) \cdot exp(- \mid \frac{e_t^k}{\sqrt{2}\sigma_k} \mid^2). \tag{7}$$

It then follows immediately from (5), (6) and (7) that

$$Prob(y_t \mid y_{t-1}, ..., y_1; Z = \theta_k) = C(\sigma_k) \cdot exp(- \mid \frac{y_t - y_t^k}{\sqrt{2}\sigma_k} \mid^2). \qquad (8)$$

The probability assumption of (7) is arbitrary, but works well in practice, as will be seen in Section 5. The parameter σ_k^2 is the variance and $C(\sigma_k)$ is a normalizing constant. Extensions for vector valued y_t and e_t^k are obvious. The posterior probability p_t^k of source $S(\theta_k)$, $k = 1, 2, ..., K$, for time $t = 1, 2, ..,$ can be computed by means of the above equations. At time t the time series is classified to the source that maximizes the posterior probability:

$$\hat{Z}_t \doteq \arg \max_{\theta_k \in \Theta} p_t^k. \qquad (9)$$

The recursion for p_t^k is obtained from (1), (4), (5), (8) and (9).

3 Incremental Credit Assignment Scheme

In this section we introduce an Incremental CRedit Assignment (ICRA) scheme to be used for time series classification. ICRA is motivated from the Bayesian scheme but it is simpler in implementation, requiring only adders and multipliers. In addition ICRA classifies as well, and sometimes better, than the Bayesian scheme, as will become obvious in Section 5. Finally, ICRA has desirable convergence properties which can be mathematically proved. Hence ICRA is an attractive alternative to Bayesian classification.To develop ICRA, start by defining

$$g(e) \doteq C(\sigma_k) \cdot exp(- \mid \frac{e}{\sqrt{2}\sigma_k} \mid^2). \qquad (10)$$

Now consider the following difference equation

$$\frac{q_t^k - q_{t-1}^k}{\gamma} = g(e_t^k) \cdot p_{t-1}^k - \left(\sum_{j=1}^{K} p_{t-1}^j \cdot g(e_t^j) \right) \cdot q_{t-1}^k, \qquad (11)$$

with initial condition ($k = 1, 2, ..., K$)

$$q_0^k > 0, \quad \sum_{k=1}^{K} q_0^k = 1. \tag{12}$$

It is clear that *if* the q_t^k's converge, in equilibrium ($q_t^k \simeq q_{t-1}^k$) we will have $q_t^k \simeq g(e_t^k) \cdot p_{t-1}^k \; / \; \sum_{j=1}^{K} g(e_t^j) \cdot p_{t-1}^j$. Since the p_{t-1}^k's in (11) are unknown, let us substitute them by the q_{t-1}^k's. After some rewriting, eq.(11) becomes

$$q_t^k = \left\{ 1 + \gamma \left[g(e_t^k) - \left(\sum_{j=1}^{K} q_{t-1}^j \cdot g(e_t^j) \right) \right] \right\} \cdot q_{t-1}^k. \tag{13}$$

Eq.(13) is the important part of the ICRA scheme. Even though we have started with a Bayesian point of view, this can now be abandoned. We consider the q_t^k's to be credit functions: the highest q_t^k gets, the most likely is $S(\theta_k)$ to be the "true" source. From eq.(13) we see that the credit fuctions q_t^k are updated in an incremental manner, similar to a steepest descent procedure. At time t the time series is classified to source $S(Z_t^*)$, where

$$Z_t^* \doteq \arg \max_{\theta_k \in \Theta} q_t^k. \tag{14}$$

Of course the use of eq.(14) requires some justification; namely we must prove that if the "true" or "best" source is $S(\theta_m)$, then q_t^m is greater than q_t^k, $k \neq m$. This justification will be provided in the next section. Namely, we will prove that the q_t^k's as given by (13) are convergent; in particular, the q_t^m associated with source $S(\theta_m)$ of highest predictive power, converges to one, while all other q_t^k's converge to zero. Therefore the credit functions q_t^k can be used for classification.

In summary, the ICRA scheme is based on equations (12), (5), (6), (10), (13), (14), which can be implemented by the recurrent, hierarchical, modular network of Fig. 1. The bottom, prediction level of the hierarchy consists of a bank of predictive modules, each one implementing a predictor of the form (5), for a specific value θ_k. Typically these modules are feedforward

neural networks (sigmoid, linear, Gaussian etc.) The top, decision level of the hierarchy consists of a module that implements (13); this module can be built from Gaussian neurons. At this point we should emphasize that within this context the Gaussian form $g(e_t^k)$ ceases to be an assumption about the statistical properties of error e_t^k and becomes a matter of design regarding the credit assignment scheme. Also, we emphasize that ICRA can be implemented using only adders and multipliers, hence implementation is simpler than that of the Bayesian scheme. Finally, it should be mentioned that implementation of the ICRA scheme requires computation of eq.(13) for k=1, 2, ..., K, which obviously scales linearly with K, the number of classes. Hence, time requirements of ICRA are $O(K)$: to handle 100 classes takes only ten times more than to handle 10 classes if the algorithm is implemented serially. It should also be noted that eq.(13) is *fully parallelizable* (see also Fig.1) resulting in $O(1)$ (constant) execution time for parallel implementation. Memory requirements are also $O(K)$, since only the current q_t^k's need to be retained at every time step. [1]

4 Convergence

We will now show that (13) has the following property: if θ_m is the "best" value of θ (i.e., source $S(\theta_m)$ best predicts the data observed) then q_t^m converges to 1 and q_t^k converges to 0 for $k \neq m$. We start with the following lemma.

Lemma 1 *If* $\sum_{k=1}^{K} q_0^k = 1$, *then* $\sum_{k=1}^{K} q_t^k = 1$ *for* $t = 1, 2, \ldots$.

Proof: Proof will be by induction. Supposing $\sum_{k=1}^{K} q_{s-1}^k = 1$, it will be shown that $\sum_{k=1}^{K} q_s^k = 1$ as well. Summing (13) over k (and using

[1] The same time and memory requirements hold for the Bayesian classifier of Section 2.

$\sum_{k=1}^{K} q_{s-1}^k = 1$) we get:

$$\sum_{k=1}^{K} q_s^k = \sum_{k=1}^{K} q_{s-1}^k + \gamma \cdot \sum_{k=1}^{K} q_{s-1}^k \cdot g(e_s^k) - \gamma \cdot \left[\sum_{j=1}^{K} q_{s-1}^j \cdot g(e_s^j)\right] \cdot \left[\sum_{k=1}^{K} q_{s-1}^k\right] \Rightarrow$$

$$\sum_{k=1}^{K} q_s^k = 1 + \gamma \cdot \sum_{k=1}^{K} q_{s-1}^k \cdot g(e_s^k) - \gamma \cdot \left[\sum_{j=1}^{K} q_{s-1}^j \cdot g(e_s^j)\right] \cdot 1 = 1. \qquad (15)$$

Since the proposition is true for $t = 0$, applying (15) repeatedly for s= 1,2, ... proves the Lemma. •

Now we can state and prove the following convergence theorem.

Theorem 2 *Define* $a_k = E(g(e_t^k))$, $k = 1, ..., K$. *Suppose* a_m *is the* unique *maximum of* a_1, ... , a_K. *If* $q_0^m > 0$, *then* $q_t^m \to 1$ *and* $q_t^k \to 0$ *for* $k \neq m$, *with probability 1.*

Remarks: First, note that $g(e_t^k)$ is a random variable, since it is a function of the error e_t^k. Assuming e_t^k to be stationary, $a_k = E(g(e_t^k))$, i.e. the expectation of $g(e_t^k)$, is time independent. Since $g(e)$ is a decreasing function of $|e|$, a large value of a_k implies good predictive performance. In this sense, a_k can be viewed as a prediction quality index and it is natural to consider as optimal the predictor m that has maximum a_m. In the course of the proof it will become clear that any function $g(|e|)$ could be used as long as $g(|e|)$ is a decreasing function of $|e|$. The theorem can be generalized to the case where there is more than one predictor that achieves maximum a_m; then the *total* posterior probability of all such predictors will converge to 1. The proof for that case is similar to the one presented here, and is omitted for economy of space. Finally, note that the credit functions q_t^k are random variables, as they depend on $y_1, y_2, ..., y_t$. Hence, q_t^k converge in a stochastic sense, in this case with probability one.

Proof: For $t = 0, 1, 2, ...$, define $\mathcal{F}_t$ to be the sigma- field generated by q_0^k and $\{e_s^k\}_{s=0}^t$, with $k = 1, ..., K$. Define by $\mathcal{F}_\infty \doteq \cup_{t=1}^{\infty} \mathcal{F}_t$. Now, q_t^k is $\mathcal{F}_t$-

measurable, for all k, t. [2] This is so because q_t^k is a function of e_t^1, ... , e_t^K and of q_{t-1}^1, ... , q_{t-1}^K. But q_{t-1}^1, ... , q_{t-1}^K are in turn functions of e_{t-1}^1, ... ,e_{t-1}^K and of q_{t-2}^1, ... , q_{t-2}^K and so on. In short, q_t^k is a function of e_1^1, ... , e_1^K, e_2^1, ... e_t^K. Hence it is clearly $\mathcal{F}_t$- measurable. Also, for $k = 1, 2, ..., K$, $t = 0, 1, 2, ...$, define $\pi_t^k \doteq E(q_t^k)$. In (13) let $k{=}m$ and take conditional expectations with respect to $\mathcal{F}_{t-1}$. For all k and t we have $E(q_{t-1}^k \mid \mathcal{F}_{t-1}) = q_{t-1}^k$, $E(g(e_t^k) \mid \mathcal{F}_{t-1}) = E(g(e_t^k)) = a_k$. In other words, $g(e_t^k)$ is independent of $\mathcal{F}_{t-1}$. This is so because we assumed the noise process $\{e_t^k\}_{t=1}^{\infty}$ to be white, hence e_t^k is independent of e_s^l, $l = 1, ..., K$, $s = 1, ..., t-1$. Finally, from Lemma 1, $\sum_{j=1}^K q_{t-1}^j = 1$, hence

$$E(q_t^m \mid \mathcal{F}_{t-1}) = \left\{ 1 + \gamma \left[a_m - \left(\sum_{j=1}^K q_{t-1}^j \cdot a_j \right) \right] \right\} \cdot q_{t-1}^m \Rightarrow$$

$$E(q_t^m \mid \mathcal{F}_{t-1}) \geq \left\{ 1 + \gamma \left[a_m - a_m \cdot \left(\sum_{j=1}^K q_{t-1}^j \right) \right] \right\} \cdot q_{t-1}^m \Rightarrow$$

$$E(q_t^m \mid \mathcal{F}_{t-1}) \geq q_{t-1}^m. \tag{16}$$

From (16) follows that $\{q_t^m\}_{t=0}^{\infty}$ is a *submartingale*. Since $0 \leq E(\mid q_t^m \mid) \leq 1$, we can use the Submartingale Convergence Theorem and conclude that, with probability 1, the sequence $\{q_t^m\}_{t=0}^{\infty}$ converges to some random variable, call it q^m, where q^m is $\mathcal{F}_\infty$- measurable. We have assumed that $q_0^m > 0$; from this, and eq.(13) it follows that for all t we have $q_t^m > 0$. From this it is easy to prove that the limit $q^m > 0$. Hence, convergence of q_t^m *does not* depend on the initial values q_0^k, $k = 1, 2, ..., K$, as long as q_0^m is greater than zero. However, we still do not know whether the sequences $\{q_t^k\}_{t=0}^{\infty}$, $k \neq m$, converge. Similarly, since $q_t^m \to q^m$, we can take expectations and obtain

[2] A sigma-field $\mathcal{F}$ generated by random variables u_1, u_2, ... is defined to be *the set of all sets of events dependent only on* u_1, u_2, ... A random variable v is said to be $\mathcal{F}$-measurable if knowledge of u_1, u_2,... completely determines v; in other words, either v is one of u_1, u_2, ... or it is a function of them: $v(u_1, u_2, ...)$. Note that the total number of u_1, u_2, ... may be finite, countably infinite or even uncountably infinite. For more details see (Billingsley, 1986).

$E(q_t^m) \to E(q^m) = \pi^m$; but we do not know whether $E(q_t^k)$ converges for $k \neq m$. However, since $\sum_{k=1}^K q_t^k = 1$ for all t, we have that $E(\sum_{k \neq m} q_t^k) = 1 - E(q_t^m) \to 1 - \pi^m$. Now, if in (13) we set $k = m$ and take the limit as $t \to \infty$, we obtain

$$q^m = \lim_{t \to \infty} \left[\left\{ 1 + \gamma \left[g(e_t^m) - \left(\sum_{j=1}^K q_{t-1}^j \cdot g(e_t^j) \right) \right] \right\} \cdot q_{t-1}^m \right]. \qquad (17)$$

Since $q^m = \lim_{t \to \infty} q_t^m > 0$, (17) implies

$$q^m = \lim_{t \to \infty} \left\{ 1 + \gamma \left[g(e_t^m) - \left(\sum_{j=1}^K q_{t-1}^j \cdot g(e_t^j) \right) \right] \right\} \cdot q^m; \qquad (18)$$

the important point is that the quantity in curly brackets *has* a limit. Since $q^m > 0$, it can be cancelled on both sides of (18); then we get

$$1 = 1 + \gamma \cdot \lim_{t \to \infty} \left[g(e_t^m) - \left(\sum_{j=1}^K q_{t-1}^j \cdot g(e_t^j) \right) \right] \Rightarrow 0 = \lim_{t \to \infty} \left[g(e_t^m) - \left(\sum_{j=1}^K q_{t-1}^j \cdot g(e_t^j) \right) \right] \Rightarrow$$

$$\lim_{t \to \infty} \left[g(e_t^m) \cdot (1 - q_{t-1}^m) \right] = \lim_{t \to \infty} \left[\sum_{j \neq m} q_{t-1}^j \cdot g(e_t^j) \right] \Rightarrow$$

(taking expectations and using the Dominated Convergence Theorem [3])

$$\lim_{t \to \infty} \left[a_m \cdot (1 - \pi_{t-1}^m) \right] = \lim_{t \to \infty} \left[\sum_{j \neq m} \pi_{t-1}^j \cdot a^j \right] \Rightarrow$$

(define $a_l \doteq \max_{k \neq m} a_k$ and note that $a_l < a_m$)

$$\lim_{t \to \infty} \left[a_m \cdot (1 - \pi_{t-1}^m) \right] \leq a_l \cdot \lim_{t \to \infty} \left[\sum_{j \neq m} \pi_{t-1}^j \right] \Rightarrow a_m \cdot (1 - \pi^m) \leq a_l \cdot (1 - \pi^m). \qquad (19)$$

[3] The Dominated Convergence Theorem states that, under appropriate conditions, $\lim_{t \to \infty} E(f_t) = E(\lim_{t \to \infty} f_t)$. See also (Billingsley, 1986).

From (19) it follows immediately that $\pi^m = 1$; otherwise we could cancel $1 - \pi^m$ from both sides of (19) and obtain $a_m \leq a_l$, which is a contradiction. Hence $1 = \pi^m = \lim_{t\to\infty} \pi_t^m$, i.e. $1 = \lim_{t\to\infty} E(q_t^m) = E(\lim_{t\to\infty} q_t^m)$. Since $\lim_{t\to\infty} q_t^m \leq 1$, we must have $\lim_{t\to\infty} q_t^m = 1$ with probability 1; it follows that $\lim_{t\to\infty} q_t^j = 0$, for $j \neq m$, which completes the proof. •

5 Examples

1. Logistic Classification. A logistic time series is produced by the following recursion (the source parameter is α)

$$x_{t+1} = \alpha \cdot x_t \cdot (1 - x_t) \qquad t = 1, 2, ...$$

1.1 In the first set of experiments, a test time series has been generated by running a logistic with α=3.8, for 182 time steps and then switching α to 3.6 and running the logistic for another 182 steps. Zero-mean white noise, uniformly distributed in the interval $[-\frac{A}{2}, \frac{A}{2}]$ has been added to the data. We have used A=0.00, 0.05, ... , 0.50. We plot the time series (at noise level A=0.2) in Fig.2. The task is to detect the active value of α. We use our ICRA scheme and compare it to the Bayesian scheme. In both cases we use the same type of predictor modules. Ten predictor modules (18-5-1 sigmoid, feedforward neural networks) have been trained on logistics with α= 3.0, 3.1, ..., 3.9, respectively. Average predictor training time was 2.5 min on a Sun Sparc IPC workstation. The σ parameter is the same for both classifiers; we take it equal to the experimentally computed standard deviation of predictor error. For all prediction modules this is approximately equal to 0.25; so we have $\sigma_1 = ... = \sigma_{10} = 0.25$. A probability threshold parameter $h = 0.01$ is also used. For the ICRA method we also use $\gamma = 0.99$. Different values of γ do not affect classification performance, as long as they are not too low. In general, small values of σ and large values of γ result in faster update of the p_t^k and q_t^k (see eq.(13)), hence in faster response of the algorithm. Finally, it should be mentioned that choice of p_0^k, q_0^k does not affect the convergence, as remarked in the previous section. This conclusion

was supported by our experiments: while we tried several values for p_0^k, q_0^k classification performance was not affected. In the experiments reported here, we have always used $p_0^k = 1/K$, $q_0^k = 1/K$.

In Fig.3 the evolution of the q_t^k's is plotted for a typical experiment. Classification to the true logistic takes very few time steps: at t=2 $q_t^9 > q_t^k$, $k \neq 9$ and at t=8 it reaches its steady state value; then at t=183 we have the α transition and by time t=189 we have $q_t^7 > q_t^k$, $k \neq 7$; at t=194 q_t^7 has reached steady state (the whole transition takes 12 time steps). Location and width of the transition points of this experiment are typical; all the classification experiments we have run gave similar results. It should be emphasized that no training is required for the decision module; its online operation only requires computation of eqs.(5), (13) for all ten predictors (k=1, ... , 10). Classification of each time step requires 0.08 sec on a Sun Sparc IPC workstation.

Classification performance is measured by dividing the number of time steps for which α is correctly identified and dividing by 364, the total number of time steps. Thus we obtain two figures of merit: one for the Bayesian and one for the ICRA method. The results, for various noise levels A are summarized in Fig.4. We see that in the noise-free case both schemes perform very well, the Bayesian scheme slightly outperforming the ICRA scheme. However, the ICRA scheme is more robust to higher noise levels.

1.2 In the second set of experiments we want to evaluate classification performance when the actual α parameter *is not* in our search set. To this end we train ten *linear* predictors on α= 3.0, 3.1, ... , 3.9 values. Training time per predictor was slightly over 1 sec on a Sun Sparc IPC workstation. Then we generate five 364-steps test logistics with an α transition at step 182. The α transitions are 3.7$-\delta\alpha$ to 3.9$+\delta\alpha$, where $\delta\alpha$ takes the values 0.00, 0.01, 0.02, 0.03, 0.04. Hence $\delta\alpha$ measures the difference between the α on which we trained our search set and the actual α value which generates the test time series. Note that for $\delta\alpha$ = 0.05 we get α= 3.65, exactly halfway between the search set α's 3.6 and 3.7. All the other parameters of the experiments are the same as in the previous paragraph. With the exception

of the first case, the *true* values of α are not in our search set. The results of these experiments are summarized in Fig. 5. Classification at a time step is considered to be correct when the time series is classified to the value of α in the search set which is closest to the true value of α. In other words, for all five time series correct classification should be: α=3.7 for the first 182 steps and α=3.9 for the last 182 steps. In Fig. 5 we plot classification figure of merit vs. $\delta\alpha$. We see that the ICRA scheme performs better than the Bayesian scheme: it is more robust to parameter variations. Of course, an additional conclusion of this experiment set is that classification can be succesfully performed using *linear* predictors. Finally, let us note that classification of each time step requires 0.04 sec on a Sun Sparc IPC workstation.

2. Enzyme Classification. This experiment involves classification of the β-lactamase enzymes. The data and problem are described in (Papanicolaou & Medeiros, 1990); here we give a short overview. β-lactamases determine resistance to β-lactam antibiotics. Classification of β-lactamases is a problem which has received considerable attention by biomedical researchers. A classification method, presented in (Papanicolaou & Medeiros, 1990) uses an "inhibition" experiment. The β-lactamase enzyme causes hydrolysis of a chemical called nitrocefin, and the β-lactam slows hydrolysis down by inhibiting the action of the enzyme. In the following paragraphs we use the terms enzyme (in place of β-lactamase) and inhibitor (in place of β-lactam). For every enzyme / inhibitor pair an "inhibition profile" is obtained, which (for a given inhibitor) characterizes the enzyme. This method has a high classification success, but the following problem occurs: the properties of enzymes and inhibitors heavily depend on the conditions under which they are prepared, and this results in varying inhibition profiles for different preparations of the same enzyme/inhibitor pair. However some *dynamic* properties of the profile remain invariant; in (Papanicolaou & Medeiros, 1990) it is reported that enzyme classification depends on the slope of the inhibition profile at various times during the experiment, as well as on the final concentration of nitrocefin. This information was used by a human operator,

who classified the enzyme by combining the various characteristics of an inhibition profile.

We use the ICRA and Bayesian schemes to automate the enzyme classification process. The inhibition profiles are used as input time series. Eight enzymes are classified. The data set of inhibition profiles is separated into a test set and a training set[4]. We use two data sets, consisting of inhibition profiles for two different inhibitors and all eight enzymes. In Fig. 6 we plot inhibition profiles for three enzymes from the training set and the same three enzymes from the test set. In all cases the same inhibitor has been used. It is noted that for the same enzyme, the test profile can differ significantly from the training profile, for the reasons explained in the previous paragraph. For each enzyme a *sixth order linear predictor* is trained on the corresponding inhibition profile from the training set. (These profiles are 40 min long time series; each time step represents 0.5 min of real time.) This is the offline training phase, which takes less than 1 sec per predictor on a Sun Sparc IPC workstation. Mean square prediction error is approximately 0.05 for all profiles. Next, we choose an inhibition profile from the test set and proceed to determine the enzyme it corresponds to. Both Bayesian and ICRA scheme are used; in Fig. 7 we present q_t^k evolution for a particular enzyme inhibition profile. In this task final classification uses values q_{40}^1, q_{40}^2 ,..., q_{40}^8 (p_{40}^1, p_{40}^2, ... , p_{40}^8, respectively). Classification performance of the Bayesian scheme is measured by c_p, the number of correctly classified enzymes (at time t=40 min) divided by eight, the total number of enzymes. A similar number, c_q, is computed for the ICRA scheme. For the Bayesian scheme we find $c_p = 0.875$, i.e. seven out of eight enzymes were correctly classified. For the ICRA scheme we find $c_q = 1.000$, i.e. all eight enzymes were correctly classified. Therefore, in this experiment the ICRA scheme classifies better than the Bayesian scheme. Classification of each time step requires 0.03 sec on a Sun Sparc IPC workstation.

[4]We want to thank G.A. Papanicolaou for kindly permitting us to use the inhibition profile data.

6 Conclusions

We have presented ICRA, an incremental credit assignment scheme for time series classification. ICRA is implemented by a recurrent, hierarchical, modular neural network which consists of a decision module and a bank of predictive modules. The decision module implements a Gaussian function $g(e)$ (where e is prediction error) but any function $g(\cdot)$ can be used, as long as it is a decreasing function of $|e|$. The predictive modules can be sigmoid, linear, Gaussian etc. feedforward networks. In fact, because of the competitive nature of the ICRA scheme, classification depends on *relative*, not *absolute* predictive performance, making ICRA robust to noise and prediction error. We have proven that, under mild conditions, ICRA converges to the correct result, i.e. it detects the time series source that best predicts the observed data. The ICRA classifier is recursive, appropriate for online time series classification which must be updated at every time step, taking into account past classification as well as the dynamic behavior of the time series. ICRA is modular and parallelizable, which means that offline training (of the predictor modules) as well as online operation scale linearly with the number of classes handled. No online training is necessary. Hence, to train and classify 100 logistics would take ten times as long as to train and classify 10 logistics; in principle there is no limit to the number of classes that can be handled. Online operation time is $O(K)$ (where K is the number of classes) for serial operation and $O(1)$ for parallel operation, i.e. all per step classification times reported in the previous section would be reduced by approximately $1/K$ if ICRA was implemented in parallel.

The above paragraph summarizes the basic features of ICRA classification. These also hold for the Bayesian classifier of Section 2. However, the experiments of Section 5 indicate that ICRA classification is more accurate and robust than Bayesian classification. In addition, unlike Bayesian, the ICRA classifier can be implemented using only adders and multipliers; hence a simple and fast hardware implementation is possible. This is a further advantage over the Bayesian classification scheme, which requires a more complicated implementation. In short, the advantages listed in this

and the previous paragraph make ICRA an attractive recursive method for time series classification problems, where past classification results must be used for future classification, and classes are given in advance.

Acknowledgements. The authors want to thank the anonymous referees for their insightful comments.

References

[ ] H.E. Ayestaran and R.W. Prager, 1993. The logical gates growing network. Cambridge Un. Engineering Dept., TR CUED F-INFENG TR 137.

[ ] W.G. Baxt, 1992. Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation*, **4**.

[ ] P. Billingsley, 1986. *Probability and Measure*. Wiley, New York.

[ ] J.D. Farmer and J.S. Sidorowich, 1988. Exploiting chaos to predict the future and reduce noise. Los Alamos Nat. Laboratory, TR LA UR 88 901.

[ ] J. Hertz, A. Krogh and R.G. Palmer, 1991. *Introduction to the Theory of Neural Computation*, Addison- Wesley, Redwood City.

[ ] C.G. Hilborn and D.G. Lainiotis, 1969. Optimal estimation in the presence of unknown parameters. *IEEE Trans. on Systems Science and Cybernetics*, **5**, 38-43.

[ ] R.A. Jacobs et al., 1991. Adaptive mixtures of local experts. *Neural Computation*, **3**, 79-87.

[ ] M.I. Jordan and R.A. Jacobs, 1992. Hierarchies of adaptive experts. In *NIPS 4*, eds. J. Moody, S. Hansen and R. Lippman, San Mateo, CA, Morgan Kauffman.

[ ] M.I. Jordan and R.A. Jacobs, 1994. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, **6**, 181-214.

[ ] V. Kadirkamanathan and M. Niranjan, 1992. Application of an architecurally dynamic neural network for speech pattern classification. *Proc. of the Inst. of Acoustics* **14**, 343-350.

[ ] D.G. Lainiotis and K.N. Plataniotis, 1994. Adaptive Dynamic Neural Network Estimation. In *Proc. of IJCNN 1994*, **6**, 4736-4745.

[ ] J. Moody, 1989. Fast Learning in multi-resolution hierarchies. Dept. of Computer Sc., Yale Un., TR YALEU DCS RR 681.

[ ] R.M. Neal, 1991. Bayesian mixture modelling by Monte Carlo simulation. Dept. of Computer Science, Un. of Toronto, TR CRG-TR-91-2.

[ ] S.J. Nowlan, 1990. Maximum likelihood competitive learning. In *NIPS 2*, ed. D. Touretzky, Morgan Kauffman, San Mateo, CA.

[ ] G.A. Papanicolaou and A.A. Medeiros, 1990. Discrimination of Extended-Spectrum β-Lactamases by a Novel Nitrocefin Competition Assay. *Antimicrobial Agents and Chemotherapy*, **34**, 2184-2192.

[ ] M.P. Perrone and L.N. Cooper, 1993. When networks disagree: ensemble methods for hybrid neural networks. In *Neural Networks for Speech and Image Processing*, ed. R.J. Mammone, Chapman-Hall.

[ ] L.R. Rabiner and R.W. Schafer, 1988. *Digital Processing of Speech Signals*. Prentice-Hall, Englewood Cliffs.

[ ] R.S. Shadafan and M. Niranjan, 1994. A dynamic neural network architecture by sequential partitioning of the input space, *Neural Computation*, **6**, 1202-1222.

[ ] H. Schwarze and J. Hertz, 1992. Generalization in a large commmittee machine. *Preprint*, The Niels Bohr Institute.

CAPTIONS

Figure 1. The Network architecture. Summation neurons are denoted by Σ. Gaussian neurons are denoted by G , identity neurons are denoted by I . The symbol $\leftarrow$ denotes weights determined by q_t^k. The block denoted DECISION MODULE implements eq.(13).

Figure 2. Plots of logistic time series: for t=1, 2, ..., 182 we have α=3.8; for t= 183, ..., 364 we have α=3.6. Noise level is A=0.2.

Figure 3. Logistic classification for ten sources (α= 3.0, 3.1, 3.2, ... , 3.9), t=1, 2 , ... , 364. The solid line corresponds to q_t^9 (α= 3.8) and the dotted line corresponds to q_t^7 (α=3.6). For $k \neq 7, 9$ q_t^k go to zero very rapidly and are not discernible in the figure.

Figure 4. Figures of merit for logistic classification at various noise levels. A denotes the noise level. Here ICRA figure of merit (respectively Bayesian figure of merit) denotes fraction of correctly classified time steps (out of a total 364) by ICRA (respectively Bayesian) scheme. We observe that while in the noise free case the Bayesian scheme performs slightly better than the ICRA scheme, ICRA is more robust to noise. (In all experiments we use h=0.01, σ=0.25, γ=0.99.)

Figure 5. Logistic classification for α outside the search set. ICRA figure of merit (respectively Bayesian figure of merit) denotes fraction of correctly classified time steps (out of a total 364) by ICRA (respectively Bayesian) scheme. This is plotted against difference $\delta\alpha$. We observe that when the actual α values are in the search set ($\delta\alpha$=0.0) the Bayesian scheme is slightly better than the ICRA scheme. However, ICRA is more robust to increased $\delta\alpha$. (In all experiments we use h=0.01, σ=0.25, γ=0.99.)

Figure 6. Enzyme inhibition profiles for enzymes 1 (solid lines), 2 (dashed lines) and 3 (dash-dotted lines). 1a, 2a, 3a are training data, 1b, 2b, 3b are test data.

Figure 7. Enzyme classification. The dotted line corresponds to the credit function q_t^1 of the correct enzyme. The solid line corresponds to overlapping plots of q_t^2, ... , q_t^8 . Classification is based on the final value of q_t^1.

Figure 2

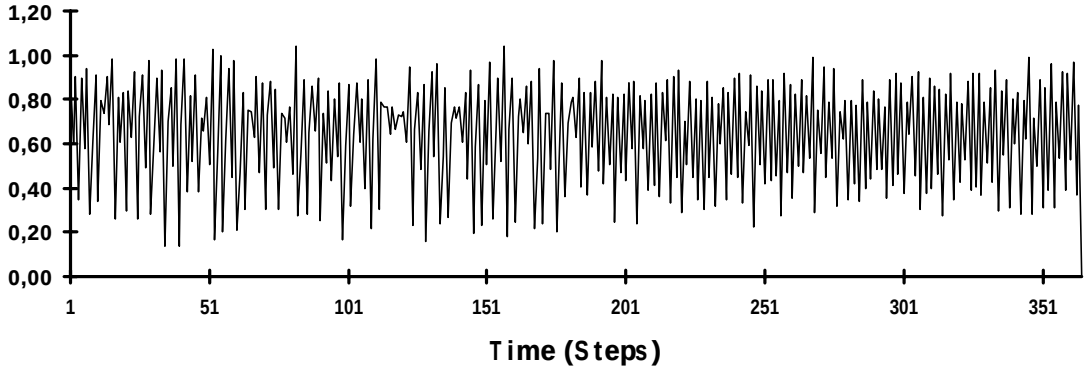

Figure 3

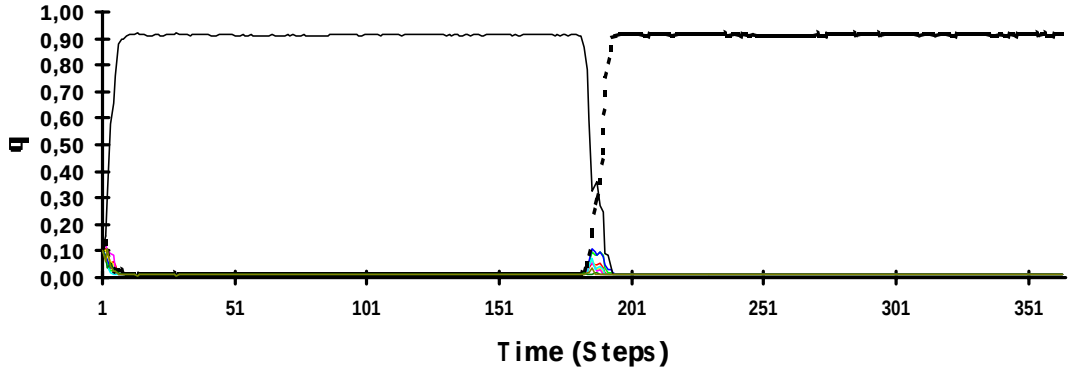

Figure 4

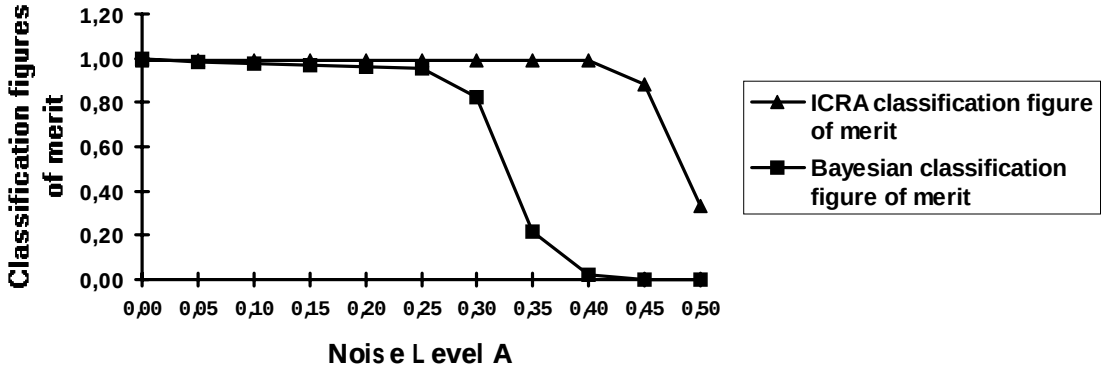

Figure 5

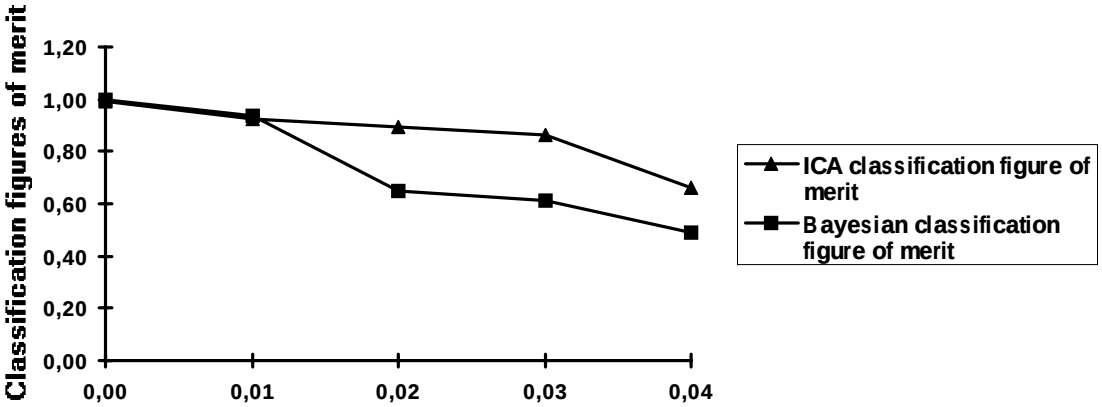

Figure 6

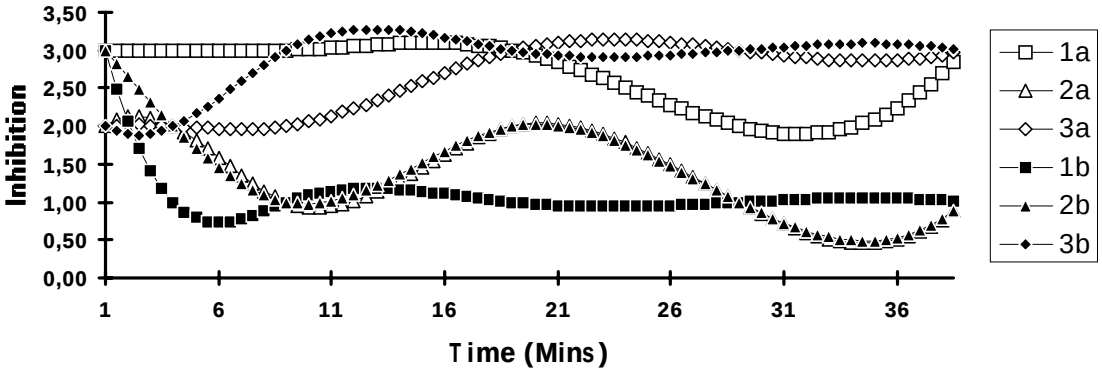

Figure 7