Pattern Recognition 43 (2010) 897–905

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.de/locate/pr

Sequential non-stationary dynamic classification with sparse feedback

D.R. Lowne, S.J. Roberts *, R. Garnett

Pattern Analysis and Machine Learning Research Group, Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK

ARTICLE INFO

Article history:
Received 20 December 2007
Received in revised form
5 August 2009
Accepted 1 September 2009

Keywords:
Non-stationary dynamic classification
Sequential Bayesian learning
Missing data
Medical signal analysis
Brain–computer interface

ABSTRACT

Many data analysis problems require robust tools for discerning between states or classes in the data. In this paper we consider situations in which the decision boundaries between classes are potentially non-linear and subject to "concept drift" and hence static classifiers fail. The applications for which we present results are characterized by the requirement that robust online decisions be made and by the fact that target labels may be missing, so there is very often no feedback regarding the system's performance. The inherent non-stationarity in the data is tracked using a non-linear dynamic classifier, the parameters of which evolve under an extended Kalman filter framework, derived using a sequential Bayesian-learning paradigm. The method is extended to take into account missing and incorrectly labeled targets and to actively request target labels. The method is shown to work well in simulation as well as when applied to sequential decision problems in medical signal analysis.

1. Introduction

An important problem in the field of online data analysis is that of adaptive classification. The non-stationarity and non-linear nature of many problem domains renders simple, static systems ineffective in forming appropriate classifiers. Employing a set of basis functions enables non-linear classifiers to be constructed and leads to the very effective radial basis function (RBF) classifiers [1]. Using a Bayesian approach to evolve a time-dependent set of parameters yields a suitably adaptive system in the presence of a non-stationary data set [2–6].

In this paper, we present an algorithm for adaptive non-linear classification that extends the non-stationary linear logistic regression system described in [4–6] to function in a non-stationary, non-linear, environment in which feedback is only sparsely available (that is, target labels are only observed on occasion), observations may be missing. We further extend the generic approach by explicit modeling of errors which may be present in the target labels. Our model may be seen as a generalized *semi-supervised* radial basis function approach.

2. Methods

2.1. Logistic regression and basis function models

We consider a decision theoretic problem in which we have available a data set of (in our case sequential) input–output pairs of the form $(\mathbf{x}, z)$ where z is an indicator, or label, target variable. We note that, in general, either of these variables, or any parts thereof, may be missing at any instant; that is, we may only have available $\mathbf{x}$ or z or neither, and in general *any* component of the observation vector $\mathbf{x}$ may be missing. For notational convenience we present the theory for a two-class decision problem with target labels $z=0$ and 1.

Our goal in this decision framework entails inferring the posterior class probability $y(\mathbf{x}) \overset{\text{def}}{=} p(z=0|\mathbf{x})$, with the probability $p(z=1|\mathbf{x}) = 1 - y(\mathbf{x})$. The approach we take is that of a *basis function model* (BFM) which has the form

$$y(\mathbf{x}) = g(\mathbf{w}^{\mathsf{T}} \boldsymbol{\varphi}(\mathbf{x})), \tag{1}$$

where $\mathbf{w}$ is a vector of parameters (weights), $\boldsymbol{\varphi}(\mathbf{x})$ a vector of responses to the input $\mathbf{x}$ via a set of non-linear kernel functions and $g(\cdot)$ is a *link function* which maps to the outcome variable y. The kernels we utilize are based on Gaussian expansions, hence our model becomes a *radial basis function* (RBF) system [1]:

$$\boldsymbol{\varphi}(\mathbf{x}) = \left\{ \begin{matrix} \mathbf{x} \\ \boldsymbol{\phi}(\mathbf{x}) \\ 1 \end{matrix} \right\}, \tag{2}$$

where the elements of the vector $\boldsymbol{\phi}(\mathbf{x}) = \{\phi_1(\mathbf{x}), \ldots, \phi_J(\mathbf{x})\}^{\mathsf{T}}$ are the responses of $\mathbf{x}$ under a set of J Gaussian kernel functions, for which the j-th element is given by

$$\phi_j(\mathbf{x}) \propto \exp\left\{ -\frac{1}{2\sigma^2}(\mathbf{x} - \boldsymbol{\mu}_j)^{\mathsf{T}}(\mathbf{x} - \boldsymbol{\mu}_j) \right\}, \quad \forall j = 1 \ldots J. \tag{3}$$

For all the results discussed in this paper, we select the J location parameters, $\boldsymbol{\mu}_j$ once, without re-adjustments, from within the statistics of $\mathbf{x}$; that is, they are placed randomly throughout the

* Corresponding author. Tel.: +44 1865 273152; fax: +44 1865 283145.
E-mail addresses: dlowne@robots.ox.ac.uk (D.R. Lowne), sjrob@robots.ox.ac.uk (S.J. Roberts), rgarnett@robots.ox.ac.uk (R. Garnett).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.09.004

effective range of the feature space.[1] Similarly, the scale parameter σ can be set to the mean distance between the kernel centers and remain fixed. The inclusion in Eq. (2) of a unity component in the basis vector allows for a level-shift in the output and the addition of the linear term, i.e. the inclusion of $\mathbf{x}$ itself, ensures that systems with a strong linear component are easily modeled.

For the two-class classification problem, the function $g(\cdot)$ may be derived via the appropriate error functional associated with distortion between y and z, that is, the model and "true" label. For a two-class decision process, the error functional is the negative logarithm of the Bernoulli distribution, which gives rise to the cross-entropy error

$$E(y,z) = -\log p(z|y) = -\log\{y^z(1-y)^{1-z}\} = z\log y + (1-z)\log(1-y). \tag{4}$$

Uniquely, to interpret the output of the BFM as a posterior probability, the function $g(\cdot)$ must be the logistic function with activation $a = \mathbf{w}^{\mathsf{T}}\boldsymbol{\varphi}$ [1]. This then gives

$$y = g(\mathbf{w}^{\mathsf{T}}\boldsymbol{\varphi}) = \frac{\exp(\mathbf{w}^{\mathsf{T}}\boldsymbol{\varphi})}{1+\exp(\mathbf{w}^{\mathsf{T}}\boldsymbol{\varphi})}. \tag{5}$$

Making predictions purely on the basis of Eq. (5), however, fails to take into account any uncertainty associated with the weight vector $\mathbf{w}$. Intuitively our predicted class label should reflect a broad distribution of the weights by making less confident statements; that is, it should tend towards a probability of 0.5 instead of 0 or 1 in a two-class problem. This principle is known as *moderation* [7,1] and is achieved using a Bayesian approach in which the uncertainty in the latent variable a is integrated into the output probability.

In taking a Bayesian approach, one entertains a distribution over the weights, $p(\mathbf{w})$, which we here assume to be a Gaussian, $\mathcal{N}(\mathbf{w};\overline{\mathbf{w}},\mathbf{P})$, with mean $\overline{\mathbf{w}}$ and covariance $\mathbf{P}$. This implies that the activation a is also Gaussian distributed with mean $\overline{a}$ and variance s^2 given by

$$\overline{a} = \overline{\mathbf{w}}^{\mathsf{T}}\boldsymbol{\varphi}, s^2 = \boldsymbol{\varphi}^{\mathsf{T}}\mathbf{P}\boldsymbol{\varphi}. \tag{6}$$

To compute the predicted class probability given $p(a) = \mathcal{N}(a;\overline{a},s^2)$ requires integrating

$$y = \int g(a)p(a)\,\mathrm{d}a, \tag{7}$$

which cannot be performed analytically. Following [7], however, we approximate the integral by

$$y \approx g(\kappa(s^2)\overline{a}), \tag{8}$$

where

$$\kappa(s^2) = \left(1+\frac{\pi s^2}{8}\right)^{-1/2}, \tag{9}$$

and we refer to y in Eq. (8) as the *moderated* output of the logistic regression function.

The effect of moderation is to scale the magnitude of large activation values down in proportion to variance (and thus the uncertainty of the estimated weights) and shift the probability value of any class label closer to 0.5 (that is, down from unity or up from zero). Note that this moderated output represents the posterior class probability, inclusive of the uncertainty intrinsic to the parameters of the model. The uncertainty, u, of the posterior probability output y may be obtained from the variance of a Bernoulli distribution. Given a posterior class probability y, the uncertainty is given by

$$u = y(1-y). \tag{10}$$

For a two-class classification problem, y can be understood as being drawn from a Bernoulli distribution with variance given by Eq. (10).

The model has a set of parameters $\mathbf{w}$ that must be globally inferred from the data. We could at this stage turn to several well-used methods for inference over the unknown parameters of the model, for example, maximum likelihood estimation. However, in order to enable such a model to handle non-stationarities in the data, we allow the vector $\mathbf{w}$ to become time-dependent and adaptive. A model of the form taken in this paper is well-suited to inference using Bayesian updates formulated in the computationally efficient framework of the *extended Kalman filter* (EKF) which is presented in the next section (see also [4]). There are numerous excellent discussions of the extended Kalman filter [8,9] and we give only a brief overview in this paper.

2.2. Dynamic basis function model for classification

We start this section with a brief note on nomenclature. The use of the subscript $t-1$ denotes variables inferred from the previous timestep before any update to the model is performed. By using the subscript $t|t-1$ we denote that the parameters for time step t have been updated by the dynamical system, before observing data at time t. With the subscript $t|t$, we denote the updated parameters for time step t after taking into account the data at time t.

Consider the dynamical system model for the posterior probability y_t of Eq. (8),

$$y_t = g(\kappa(s_t^2)\overline{a}_t), \tag{11}$$

where $\overline{a}_t$ and s_t^2 are the mean and variance of a_t, respectively. We may re-write this activation, a_t, in the form

$$a_t = \overline{\mathbf{w}}_t^{\mathsf{T}}\boldsymbol{\varphi}_t + n_t, \tag{12}$$

in which n_t is a zero-mean noise process with variance r_t^2; $\boldsymbol{\varphi}_t$ are, as before, the set of basis responses to observed input $\mathbf{x}_t$; and $\overline{\mathbf{w}}_t$ are the expectations of the time-dependent parameters of the model.

Similarly, consider a dynamical system for the time evolution of $\mathbf{w}_t$, namely

$$\mathbf{w}_{t|t-1} = f(\mathbf{w}_{t-1},\mathbf{v}_t) \tag{13}$$

in which $\mathbf{v}_t$ is a state (or parameter) noise term and $f(\cdot)$ encodes any known deterministic dynamics in the evolution of $\mathbf{w}$. In our applications we do not assume deterministic dynamics and hence the above equation is simplified to a diffusion step

$$\mathbf{w}_{t|t-1} = \mathbf{w}_{t-1} + \mathbf{v}_t. \tag{14}$$

We use a Gaussian prior distribution for the weights $\mathbf{w}$ with mean $\overline{\mathbf{w}}_{t-1}$ and variance $\mathbf{P}_{t-1}$; that is,

$$\mathcal{N}(\mathbf{w}_{t-1};\overline{\mathbf{w}}_{t-1},\mathbf{P}_{t-1}). \tag{15}$$

The aim of the Kalman update algorithm is to compute the updated prior distribution of the parameters, $\mathcal{N}(\mathbf{w}_{t|t-1};\overline{\mathbf{w}}_{t|t-1},\mathbf{P}_{t|t-1})$, and thence to infer the distribution over the posterior parameters, $\mathcal{N}(\mathbf{w}_{t|t};\overline{\mathbf{w}}_{t|t},\mathbf{P}_{t|t-1})$.

Following the standard formulation for the EKF we start by defining

$$\mathbf{F}_t = \left.\frac{\partial f}{\partial \mathbf{w}}\right|_{\mathbf{w}=\overline{\mathbf{w}}_{t|t-1},\mathbf{v}=\overline{\mathbf{v}}}, \quad \mathbf{G}_t = \left.\frac{\partial g}{\partial \mathbf{w}}\right|_{\mathbf{w}=\overline{\mathbf{w}}_{t|t-1},n=\overline{n}},$$
$$\mathbf{J}_t = \left.\frac{\partial f}{\partial \mathbf{v}}\right|_{\mathbf{w}=\overline{\mathbf{w}}_{t|t-1},\mathbf{v}=\overline{\mathbf{v}}}, \quad \mathbf{V}_t = \left.\frac{\partial g}{\partial n}\right|_{\mathbf{w}=\overline{\mathbf{w}}_{t|t-1},n=\overline{n}}, \tag{16}$$

where the functions g and f are defined in Eqs. (11)–(14) above and $\overline{\mathbf{v}}$ and $\overline{n}$ are the means of the state- and observation-noise components,

[1] We do not consider online re-adjustment in this paper because, in our target applications, the range of the feature space may be determined ahead of time without loss of generality.

respectively. We define the observation and state noise variances as

$$\mathbf{R}_t = r_t^2, \quad \mathbf{Q}_t = \mathrm{cov}[\mathbf{v}_t]. \tag{17}$$

Evaluating the above expressions for our model gives

$$\mathbf{F}_t = \mathbf{I}, \quad \mathbf{G}_t = y_{t|t-1}(1-y_{t|t-1})\boldsymbol{\varphi}_t^{\mathrm{T}}, \quad \mathbf{J}_t = \mathbf{I}, \quad \mathbf{V}_t = \mathbf{1}, \tag{18}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{1}$ a column vector of ones.

The updated priors are formulated such that the parameters of the model are adjusted before observing the data at time step t. The mean and variance of the parameters are adjusted according to

$$\overline{\mathbf{w}}_{t|t-1} = f(\overline{\mathbf{w}}_{t-1}, \mathbf{0}) = \overline{\mathbf{w}}_{t-1} \tag{19}$$

and

$$\mathbf{P}_{t|t-1} = \mathbf{F}_t\mathbf{P}_{t-1}\mathbf{F}_t^{\mathrm{T}} + \mathbf{J}_t\mathbf{Q}_t\mathbf{J}_t^{\mathrm{T}} = \mathbf{P}_{t-1} + \mathbf{Q}_t. \tag{20}$$

In the EKF, the posterior distribution for $\mathbf{w}_{t|t}$ is inferred via the definition of the *Kalman gain*, which is defined as

$$\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{G}_t^{\mathrm{T}}(\mathbf{V}_t\mathbf{R}_t\mathbf{V}_t^{\mathrm{T}} + \mathbf{G}_t\mathbf{P}_{t|t-1}\mathbf{G}_t^{\mathrm{T}})^{-1} = \frac{\mathbf{P}_{t|t-1}u_{t|t-1}}{\mathbf{R}_t + u_{t|t-1}^2 s_{t|t-1}^2}\boldsymbol{\varphi}_t, \tag{21}$$

where, as before, the uncertainty in $y_{t|t-1}$ is defined as

$$u_{t|t-1} = y_{t|t-1}(1-y_{t|t-1}), \tag{22}$$

$$y_{t|t-1} = g(\kappa(s_{t|t-1}^2)\overline{\mathbf{w}}_{t|t-1}^{\mathrm{T}}\boldsymbol{\varphi}_t), \tag{23}$$

and

$$s_{t|t-1}^2 = \boldsymbol{\varphi}_t^{\mathrm{T}}\mathbf{P}_{t|t-1}\boldsymbol{\varphi}_t. \tag{24}$$

Following [4,5] we set $\mathbf{R}_t = r_t^2 = u_{t|t-1}$, which simplifies the Kalman gain to

$$\mathbf{K}_t = \frac{\mathbf{P}_{t|t-1}}{1 + u_{t|t-1}s_{t|t-1}^2}\boldsymbol{\varphi}_t. \tag{25}$$

After observing the data ($\mathbf{x}_t$ and z_t) at time t, we update the mean and covariance of the posterior parameters according to

$$\overline{\mathbf{w}}_{t|t} = \overline{\mathbf{w}}_{t|t-1} + \mathbf{K}_t(z_t - y_{t|t-1}), \tag{26}$$

$$\mathbf{P}_{t|t} = \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{G}_t\mathbf{P}_{t|t-1}^{\mathrm{T}} = \mathbf{P}_{t|t-1} - \mathbf{K}_t u_{t|t-1}(\mathbf{P}_{t|t-1}\boldsymbol{\varphi}_t)^{\mathrm{T}}. \tag{27}$$

As we advance to the next time step, we simply designate the current posterior as the prior for the next time step; that is, in the above equations we allow

$$\overline{\mathbf{w}}_{t-1} \leftarrow \overline{\mathbf{w}}_{t|t}, \tag{28}$$

$$\mathbf{P}_{t-1} \leftarrow \mathbf{P}_{t|t}. \tag{29}$$

In this manner the EKF moves recursively through the data.

2.3. State noise inference

The state noise vector $\mathbf{v}_t$ is assumed to follow the distribution $\mathcal{N}(\mathbf{v}_t; \mathbf{0}, q_t\mathbf{I})$, where $\mathbf{I}$ is the identity matrix and $\mathbf{0}$ a column vector of zeros. In other words, we add a time-dependent, isotropic covariance matrix $q_t\mathbf{I}$ to the prior covariance matrix $\mathbf{P}_{t-1}$ to arrive at the updated weight prior at time t, $\mathcal{N}(\mathbf{w}; \overline{\mathbf{w}}_{t|t-1}, \mathbf{P}_{t|t-1})$, that takes the state noise q_t into account:

$$\overline{\mathbf{w}}_{t|t-1} = \overline{\mathbf{w}}_{t-1}, \mathbf{P}_{t|t-1} = \mathbf{P}_{t-1} + q_t\mathbf{I}. \tag{30}$$

The covariance expansion $\mathbf{P}_{t-1} + q_t\mathbf{I}$ captures the drift assumed to be present in the decision boundaries.

So far the weight noise variance q_t has not been specified. As was shown in [4], weight noise variance is instrumental in quantifying the algorithm's learning rate, which is given by $K^{-1}\mathrm{Trace}(\mathbf{P}_t)$ where K is the dimensionality of $\mathbf{P}_t$.

To guide the update of the weight noise variance, we examine the discrepancy between the predicted class uncertainty after and before observing the true class label. In particular we are looking at the information gained from the most recent update

$$\mathcal{I}_t = u_{t|t} - u_{t|t-1}, \tag{31}$$

where $u_{t|t-1}$ is given in Eq. (22) and represents the uncertainty in the class probability predicted from the input $\mathbf{x}_t$ alone, whereas $u_{t|t}$ captures the uncertainty remaining after incorporating the class label information z_t. Specifically, the posterior class uncertainty $u_{t|t}$ is the variance of the posterior class probability $y_{t|t}$ that incorporates the information from the label z_t through the posterior weight distribution $\mathcal{N}(\mathbf{w}_{t|t}; \overline{\mathbf{w}}_{t|t}, \mathbf{P}_{t|t})$. Thus, $u_{t|t}$ is defined by

$$u_{t|t} = y_{t|t}(1-y_{t|t}), \tag{32}$$

where

$$y_{t|t} = g(\kappa(s_{t|t}^2)\overline{a}_{t|t}). \tag{33}$$

Given z_t, the time-dependent state noise variance q_t is calculated as

$$q_t = \max\{\mathcal{I}_t, 0\} + z_t(1-z_t). \tag{34}$$

We therefore diffuse our decision only if we suffer an increase in uncertainty after having been informed of the true class label z_t. Note that the second term of Eq. (34) is zero if either $z_t = 0$ or $z_t = 1$ i.e. the label has no uncertainty associated with it. As we develop in the following sections, this will not always be the case and hence labeling uncertainty becomes fed back into our state noise variance term. Otherwise, we take no action and leave q_t set to its previous value. This approach is similar to that advocated in [4] but avoids the cost of explicit line searches. We note that more computationally demanding approaches may be used, such as the *variational Bayes* approach to the Kalman filter presented in [6]. For real-time applications, however, the computational simplicity of the approach described here is advantageous.

2.4. Missing data

2.4.1. Missing labels: sparse targeting

In many application domains, the incoming data are relatively inexpensive to acquire but feedback, in the form of "true" target labels, is sparse. We are required nevertheless to make a decision; to this end, we modify the adaptive correction step in Eqs. (26) and (27) to account for both targeted and non-targeted observations. We note that even when we do not have complete input–output pairs, we do not wish to discard information contained within the input vector $\mathbf{x}_t$, such as slow drift.

When class labels are not known, we may infer the missing label via the dynamic model. In this case we use

$$\hat{z}_t = p(z_t = 0|\boldsymbol{\varphi}_t) = y_{t|t-1} \tag{35}$$

as a "quasi-target" in the place of z_t in Eq. (26). The logistic update then proceeds as before. By treating the quasi-targets as if they were true, we risk the algorithm becoming excessively confident in its predictions. We note that the second term in Eq. (34) will not now be equal to zero, as $\hat{z}_t$ is in general not 0 or 1. We update q_t using

$$q_t = \max\{\mathcal{I}_t, 0\} + \hat{z}_t(1-\hat{z}_t). \tag{36}$$

We thereby ensure that the classifier does not become overly confident too quickly based on fictitious feedback.

2.4.2. Missing inputs

The situation in which the data inputs $\mathbf{x}_t$ are missing is well addressed in [10] and can be solved by setting the missing input values to zero. As a result, the corresponding components of the Kalman gain become zero and the updates to components of **w** and **P** in Eqs. (26) and (27) are canceled. As none of our candidate applications suffers (at present) from spurious missing inputs, we present this section for completeness only.

2.5. Bit errors in target labels

In many applications, the labels z cannot be expected to always represent the ground truth. The labeling process could be corrupted by a number of factors, including errors during communication and data entry. In a situation with many errors in the label stream, the dynamic classifier (DC) method described above can suffer from overcompensation and overly drastic changes to the weight parameter **w**.

To illustrate, assume the DC has developed the ability to make reasonably confident decisions, that is, decisions with y near 0 or 1. In this case, given an incorrect label z_t, the innovation residual in Eq. (26), namely $(z_t - y_{t|t-1})$, would be near 1. Correspondingly, the weights **w** experience a very large shift, and as a result the decision boundary moves drastically. This behavior should be expected when the labels can be guaranteed to be correct; after all, if we are very confident about a decision that turns out to be incorrect, something must be wrong with our classifier. When the truth of the labels cannot be assured, however, we should tread more carefully and temper our decisions by our uncertainty in the labels. We propose a novel modification to the dynamic classification model accordingly, as detailed in the next section.

2.5.1. A simple noise model and its consequences

Let us adopt a simple model for the noise present in the labeling process. For the present discussion, let $\tilde{z}_t$ represent the (possibly incorrect) observed label, and let z_t represent the true, uncorrupted label. We assume the probability that an observed label is incorrect is independent of the data observed and fixed throughout time:

$$p(\tilde{z}_t \neq z_t|\boldsymbol{\varphi}_t, t) = p(\tilde{z}_t \neq z_t) = \rho. \tag{37}$$

This assumption may not always be true, but in many important examples (such as errors introduced by a noisy communication channel), it will be valid. In many other cases, it can serve as a useful substitute for a more complicated model. To simplify the present discussion, we assume that the probability ρ is known *a priori*; of course, in almost any situation this will not be true. We will present an online method for estimating ρ from the data stream presently.

In the presentation of the dynamic classifier, we assumed that the label z_t was determined from a Bernoulli distribution with parameter y_t:

$$p(z_t|y_t) = y_t^{z_t}(1-y_t)^{(1-z_t)}. \tag{38}$$

Under our label noise model, this assumption is violated. Instead, we may derive the distribution of $\tilde{z}_t$ given y_t and ρ:

$$p(\tilde{z}_t|y_t, \rho) = (1-2\rho)(y_t^{\tilde{z}_t}(1-y_t)^{(1-\tilde{z}_t)}) + \rho. \tag{39}$$

Recalling that we defined $y_t \overset{\text{def}}{=} p(z_t = 0|\mathbf{x})$, the above discrepant distribution may be reconciled by replacing y_t with an appropriately modified output $\tilde{y}_t$, given by

$$\tilde{y}_t \overset{\text{def}}{=} (1-2\rho)y_t + \rho. \tag{40}$$

With this definition, we now observe that

$$p(\tilde{z}_t|\tilde{y}_t) = \tilde{y}_t^{\tilde{z}_t}(1-\tilde{y}_t)^{(1-\tilde{z}_t)}, \tag{41}$$

that is, the original relation between the output of the classifier and the observed label is now preserved. Under the assumed noise model, the only required modification to the dynamic classifier framework is to replace the original output of the model y_t (which assumes no noise in the labels) with the value $\tilde{y}_t$ given above. The definition of $\tilde{y}_t$ serves to moderate the certainty of our classifications according to our uncertainty in the label accuracy.

2.5.2. Estimating ρ

As mentioned above, the true value of ρ will almost certainly not be known. Fortunately, we may estimate the true value of ρ online from the data. Given our assumptions, a simple calculation shows

$$p(|\tilde{z} - y| > (1-\varepsilon)|y, \rho) = [\rho + \min(y, 1-y)][I(\min(y, 1-y) < \varepsilon)], \tag{42}$$

where $I(\cdot)$ is the indicator function. We marginalize out the nuisance parameter y:

$$p(|\tilde{z} - y| > (1-\varepsilon)|\rho) = \int p(|\tilde{z} - y| > (1-\varepsilon)|y, \rho)p(y)\,\mathrm{d}y. \tag{43}$$

Applying Eq. (42), we may separate this integral in a convenient manner:

$$p(|\tilde{z} - y| > (1-\varepsilon)|\rho) = \rho\int_0^{\varepsilon} p(y)\,\mathrm{d}y + \int_0^{\varepsilon} \min(y, 1-y)p(y)\,\mathrm{d}y. \tag{44}$$

We may approximate this integral with a Riemann sum:

$$p(|\tilde{z} - y| > (1-\varepsilon)|\rho) = \int p(|\tilde{z} - y| > (1-\varepsilon)|y, \rho)p(y)\,\mathrm{d}y \tag{45}$$

$$\approx \sum_{k=1}^{N}\left(\rho + \frac{k\varepsilon}{N}\right)p\left(\min(y, 1-y) < \frac{k\varepsilon}{N}\right). \tag{46}$$

This result may be understood as separating our misclassifications

$$p(|\tilde{z} - y| > (1-\varepsilon)|\rho) \tag{47}$$

into the errors expected given the uncertainty in our predictions

$$\int_0^{\varepsilon} \min(y, 1-y)p(y)\,\mathrm{d}y \tag{48}$$

and the errors expected given the uncertainty present in the labeling process

$$\rho\int_0^{\varepsilon} p(y)\,\mathrm{d}y. \tag{49}$$

Let us consider the significance of Eq. (43). Under our assumptions, the left-hand side, $p(|\tilde{z} - y| > (1-\varepsilon)|\rho)$, can be estimated from the data without knowledge of ρ by simply keeping a tally of the portion of previously observed labels $\tilde{z}$ for which $|\tilde{z} - y| > (1-\varepsilon)$. Additionally, everything in the right-hand side (except the unknown parameter ρ) can be determined by keeping an online estimate of the distribution of classifier outputs y assigned; a simple histogram suffices. This knowledge allows us to easily solve for ρ algebraically. Thus, with very little additional storage requirements (of the order of a few hundred bytes), we may effectively solve for the unknown parameter ρ.

The above discussion provides a simple modification to the model framework that effectively handles label noise with no additional parameters and very little additional computational overhead. Even when the labels are known to be correct, the above formulation may still be effectively used; the estimated parameter ρ will quickly assume a small value accordingly.

3. Active label requesting

When the dynamic classifier infers the predicted class label probability, $y_{t|t-1}$, we may evaluate the uncertainty associated with a decision into the maximum posterior class, as given by Eq. (22), $u_{t|t-1} = y_{t|t-1}(1 - y_{t|t-1})$. If we have no target labels (or indeed no incoming observations) then our intrinsic uncertainty increases with each time step. Defining a threshold on the uncertainty, $u_{t|t-1}$, is equivalent to placing a threshold, θ say, on the maximum posterior probability; hence if

$$\max\{y_{t|t-1}, 1-y_{t|t-1}\} < \theta \quad (50)$$

then we may actively request a target label. Results for this process are shown in Sections 4.2.1 and 4.3.4 later in this paper. In all the results presented in this paper, a threshold of $\theta = 0.9$ was used.

4. Results

4.1. Initialization

In all the experiments reported here, we initialized all components of $\overline{\mathbf{w}}$ to zero and both $\mathbf{P}$ and $\mathbf{Q}$ to the identity matrix.

4.2. An illustrative example

As a simple example of a drifting non-stationary system, we consider two overlapping Gaussian distributions rotating in a circular fashion around a central point at [0, 0], with the two distributions out of phase by π radians. The Bayes error associated with the data at any instant in time was 4%. We wish to adapt a discriminant boundary that captures the non-stationarity in the system without relying on constant feedback. In this instance we know that the underlying boundary is linear and for presentation of illustrative results we therefore ran our model with no basis functions; that is, only the bias and linear terms were included in Eq. (2). Fig. 1 shows snapshots of the decision boundary and the last 50 data points for $t = 50$ and 300. When presented with a full set of labels z, the dynamic classification (DC) model achieved a performance of near 96%, that is, near the Bayes error. A static logistic basis function classifier trained on the same data achieved only 50% accuracy due to the rotating nature of the problem. We note that as the algorithm operates in a causal, sequential manner all performance figures are for out of sample data, i.e. data which have *not* been seen by the algorithm.

We then successively removed label information. The performance with 50% labeling was hardly degraded, achieving 95% accuracy; with only 20% class labels, the classifier still achieved a 91% accuracy. The performance of the classifier was further investigated on this data with variation in fractions of observed labels from 0% to 100%. Fig. 2 shows that the variation in performance is very robust to large numbers of unobserved labels.

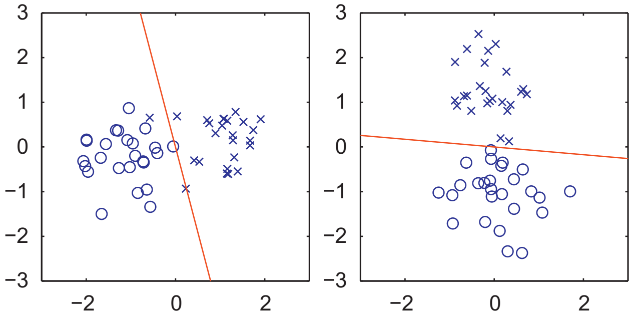

Fig. 1. Simple rotating system: plots of dynamic decision boundary and last 50 data points at $t = 50$ (left) and $t = 300$ (right) samples. Fifty percent of labels were observed. Circles and crosses denote the true class labels. The intrinsic Bayes error in this synthetic data was 4%.

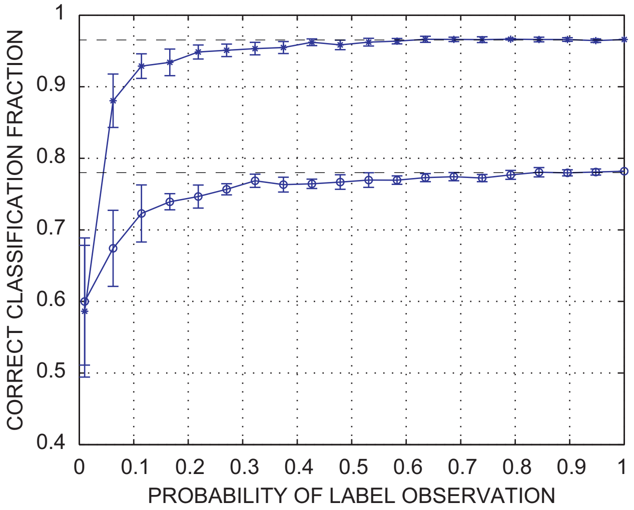

Fig. 2. Rotating data: the accuracy of the adaptive classifier remains high even in the absence of a large fraction of labels. The performance was evaluated over 10 runs with a set proportion of labels missing at random within the data stream. The plot shows the mean accuracy as a fraction of correct classifications with one standard deviation error bars. The top curve (*) is on a two-class problem with 4% Bayes error and the lower trace (o) on a noisy version with 22% Bayes error as indicated by the dashed horizontal lines in the plot. Note that in both cases the performance reaches close to optimal even with a small fraction of labeled data points. The base performance with no observed labels is of course 0.5 and is omitted for clarity. The *x*-axis represents the probability of label observation and the *y*-axis the resultant correct classification rate.

4.2.1. Active label requests

Fig. 3 shows the label request process operating with a simple static classification data set, with a linear boundary. The choice of such a simple data set is motivated by presentation simplicity rather than a necessity of the algorithm, which is generic and operates in non-stationary, non-linear environments in an identical manner. The left subplot shows the resultant boundary along with the class labels (open circles and crosses) and points for which class labels were requested (filled circles). The right-hand subplot shows the time course of the posterior class probability along with vertical bars at points when a label was actively requested.

4.2.2. The effect of label error inference

The effect of label errors becomes increasingly important in decision problems with high Bayes error (that is, intrinsic overlap between classes). This error may be due to incorrect labeling or to noise in the observation space, meaning that classes are highly overlapping. In either case we do not want to adapt the dynamic model in cases where it is believed that a datum has crossed a decision boundary due to noise. Fig. 4 shows the evolution with samples of the inferred target label noise, ρ, for the simple rotating system previously considered. The Bayes error is 22%. Fig. 5 shows the effect of taking this into account in our algorithm. Plots (a) and (b) show the parameters $\mathbf{w}_t$ for the two cases. We note that the plot in (b) follows the rotating boundary considerably better. Plots (c) and (d), respectively, show the maximum posterior probability with no inferred label noise (c) and taking it into account (d). We see that the posterior probability trace is reduced in the latter plot indicating the algorithm has taken this extra uncertainty into account.

4.3. Brain–computer interface

The goal of *brain–computer interfaces* (BCIs) is to classify observations of, generally, the brain's electrical activity during

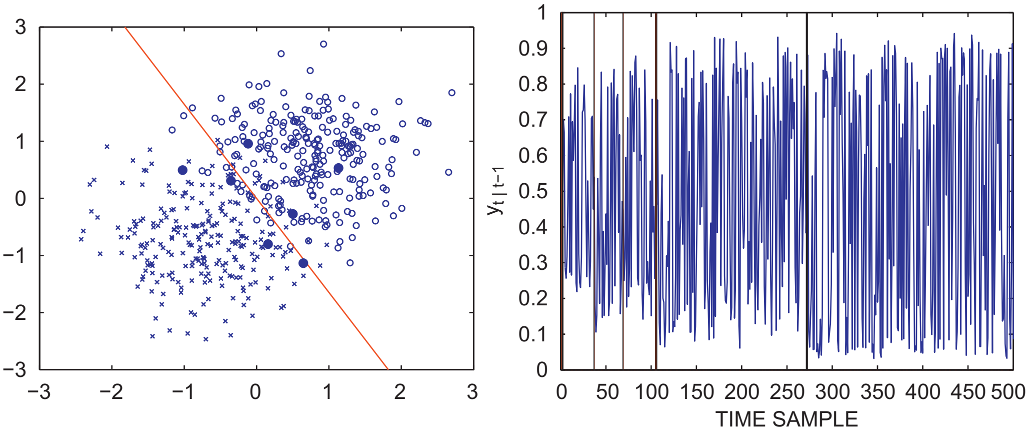

Fig. 3. Active label requesting: (left) decision boundary and requested label points (solid circles) for simple static data set—open circles and crosses represent the true class labels. (Right) time course of posterior class probability $y_{t|t-1}$ and points at which label information was requested—solid vertical bars.

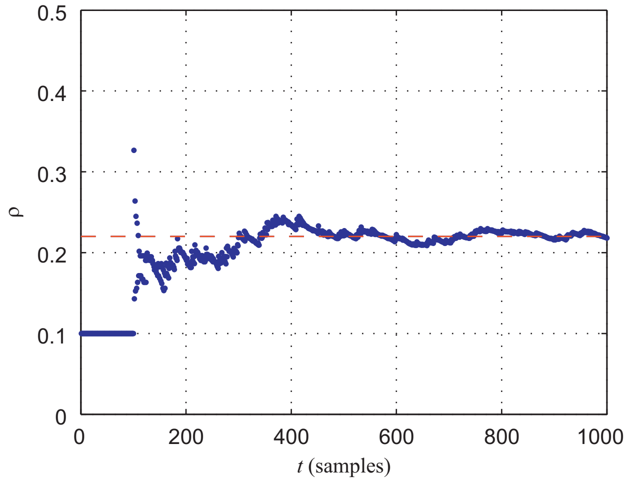

Fig. 4. Labeling error: inferred label noise, ρ. The true Bayes error was 22% as indicated by the dashed horizontal line in the figure. The inferred sample value, obtained from the estimates from time 500 onwards was $\rho = 0.2205 \pm 0.005$. The x-axis represents time in samples.

different cognitive states into one of a number of classes. The cognitive state of the subject is under (semi-) voluntary control, and therefore these decisions may then be used to drive a rudimentary computer interface.

Research into experimental, technical, and mathematical techniques for BCI has received much attention over the past few years and it is not the place of this paper to review this in detail. Review material may be found in [11]. Recent research has emphasized the need for adaptive classification [6,12].

We present results applying the methodology developed in this paper to an online BCI experiment in which the aim was to classify electroencephalogram (EEG) activity into one of two states, "movement" and "non-movement," during continuous recording.

4.3.1. *Experimental protocol*

Eight healthy subjects were recruited and asked to perform a wrist-extension exercise at semi-regular intervals cued from a computer screen. The classifier was left blind to cues, and was able to obtain feedback (that is, a true label) derived from muscle activity data in only about 20% of samples. The poverty of feedback can be explained by the fact that in only a fraction of recorded data was unambiguous label information available causally.

In this instance, the classifier was designed simply to distinguish between movement and non-movement. Thus, no distinction was made between left-hand movements and right-hand movements.

4.3.2. *Data acquisition*

Data used in this experiment consisted of two channels of EEG, recorded at 256 Hz over C3 and C4 in the standard 10–20 placement system (see for example [13] for further information regarding EEGs), and one channel of muscle electrical activity (EMG), recorded at 1024 Hz over the right flexor carpi radialis muscle. The EMG was then down-sampled to 256 Hz and muscle contraction strength for movement and non-movement detection was evaluated via a simple windowed peak and trough detection, and was causal, that is, did not look at future information. Some 20% of the data was able to be robustly labeled in this manner, leaving 80% unlabeled data.

4.3.3. *Feature extraction and basis formation*

The first and second reflection coefficients of a second-order autoregressive (AR) model [14] were calculated over each EEG signal once every 78 ms using a sliding one-second-long window, forming a set of feature vectors $\mathbf{x}_t$. These vectors were projected into a non-linear latent space via Eq. (2) using a set of five Gaussian basis functions, providing the stream of $\boldsymbol{\varphi}_t$. The choice of five basis functions was chosen by Gaussian mixture modeling of out-of-sample data not appearing in the results presented here.

4.3.4. *BCI performance*

Evaluation of the system's efficacy was performed by examining posterior class probabilities, estimated causally from the data, in conjunction with all labels retrospectively (that is, anti-causally) extracted from the EMG signals. The EEG data were analyzed sequentially using the dynamic classification algorithm. At each timeslice the posterior class probabilities, $y_{t|t-1}$, were calculated, and the corresponding class, $c_t = \text{argmax}(y_{t|t-1}, 1 - y_{t|t-1})$, was assigned. These class decisions were later compared with the targets z_t obtained from the EMG signal. Note that, as before, the classification is performed in a sequential manner, all results are by definition on out of sample data in that the algorithm utilizes past data to form a forecast and it is the decision performance of this forecast that we evaluate.

To make explicit comparison with other methods we run a static radial basis function model (using the exact same functions as the dynamic version discussed in this paper), denoted as static classifier (SC), and a standard multi-layer perceptron (MLP)

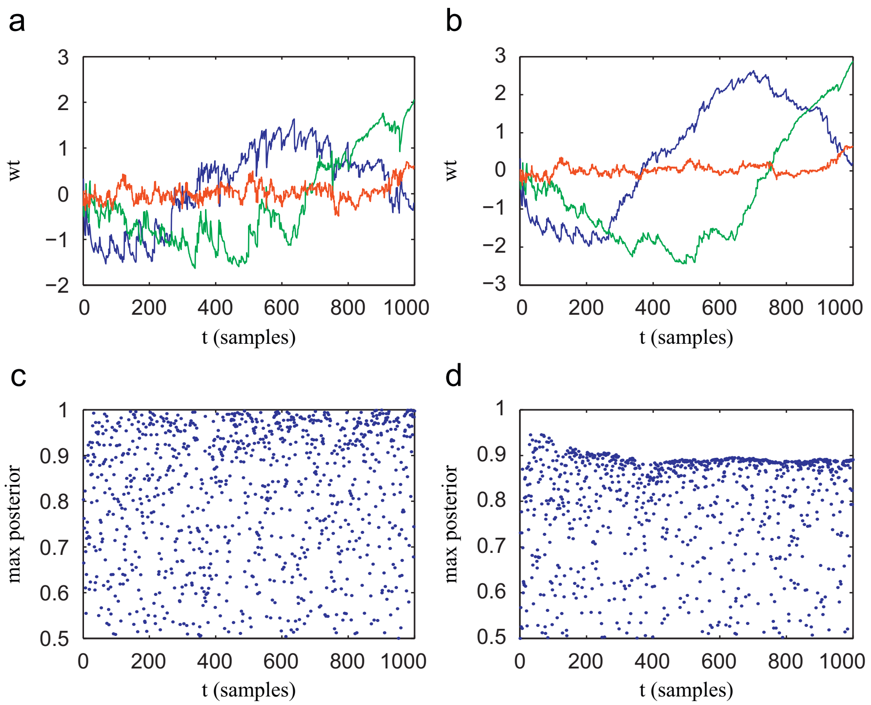

Fig. 5. Effect of label errors: (a) time course of parameters, $\mathbf{w}_t$, without label noise inference; (b) time course of parameters after inference of label noise; (c) maximum posterior class probability without label error inference; (d) maximum posterior class probability taking label noise into account.

Table 1
Accuracy of the dynamic basis function classification model (DC), the static basis function model (SC) and the multi-layer perceptron (MLP) over eight experimental runs.

Method	expt. 1	expt. 2	expt. 3	expt. 4	expt. 5	expt. 6	expt. 7	expt. 8
DC	0.982	0.982	0.987	0.994	0.992	0.997	0.980	0.984
SC	0.596	0.614	0.715	0.922	0.846	0.956	0.553	0.705
MLP	0.650	0.620	0.718	0.944	0.910	0.965	0.616	0.707

The DC was highly significantly better than either the SC or DC in all experiments ($p < 0.01$ using a Wilcoxon rank test).

classifier with logistic latent and output link functions [1]. The number of hidden-layer units in the MLP was set to two, as determined by cross-validation on a section of out-of-sample data. In both these comparative cases (SC and MLP) we trained the models on the fraction of data for which labels were available and tested their performance over all available data.

The fraction of correct classifications of the differing methods are compared in Table 1. The number of data points in the eight experiments varied from 19,000 to 56,000. The DC was highly significantly better with $p < 0.01$ in all cases using a Wilcoxon rank test. Fig. 6 shows a short section of data with posterior probability of movement for both the DC and MLP classifiers. The true target labels are also shown, as derived retrospectively from the muscle activity of the subject.

Retrospectively we also ran the dynamic classifier with active label requesting over the same data. The performance of the dynamic classifier (DC) is compared, once more, to that of a static classifier (SC) using the same basis functions as the dynamic version and also a MLP. Performance is shown in Table 2, along with an indication of the percentage fraction of data for which a label was requested (and obtained from the muscle activity trace). The SC and MLP were trained using these labeled data points only and then tested on the full data set. In all cases the DC significantly outperforms the other methods (with $p < 0.01$ once more). It is interesting to note that for the experiments for which all methods had high accuracy, e.g. expts. 4 and 6, the percentage requested labels was lowest. Once more we note that the DC acts in a causal sequential manner so results are always on unseen data. Fig. 7 shows a short section of data indicating the posterior class probability trace for the DC along with bars indicating points at which a target label was requested.

5. Discussion

In the bio-signal analysis application presented in this paper, keeping fixed basis-function expansions is ideal since the characteristics of the features being classified are well defined and tightly bounded above and below. Specifically, the feature vectors $\mathbf{x}_t$ are reflection coefficients from an AR model, thus values are bounded between −1 and 1. In a more general case, however, such a precise range and distribution of feature values may not be known although we would argue that most problems can be scaled such that inputs lie in a pre-determined range. We note that previous research has suggested dynamic readjustment of the basis functions [6].

In the current implementation, the number of basis functions, governing the dimensionality of $\boldsymbol{\varphi}_t$, was kept constant at all times. In principle, this number can be pruned, for instance, by using explicit weight-decay (ridge regression) priors [2] or Markov chain Monte Carlo approaches [15].

Future work in this area considers alternate approximate Bayesian solutions to state-space models with non-linear link functions, notably the unscented Kalman filter [16]. The work described in this paper considers only a binary decision problem. If the number of classes is more than two, then one can use a binary decision tree [17] or recast the problem in terms of a multinomial rather than binomial distribution [2].

As with all classifiers of this type, which act by forming (in this case dynamic) discriminant boundaries in the input space, care needs to be exerted in the representation of the input. The present

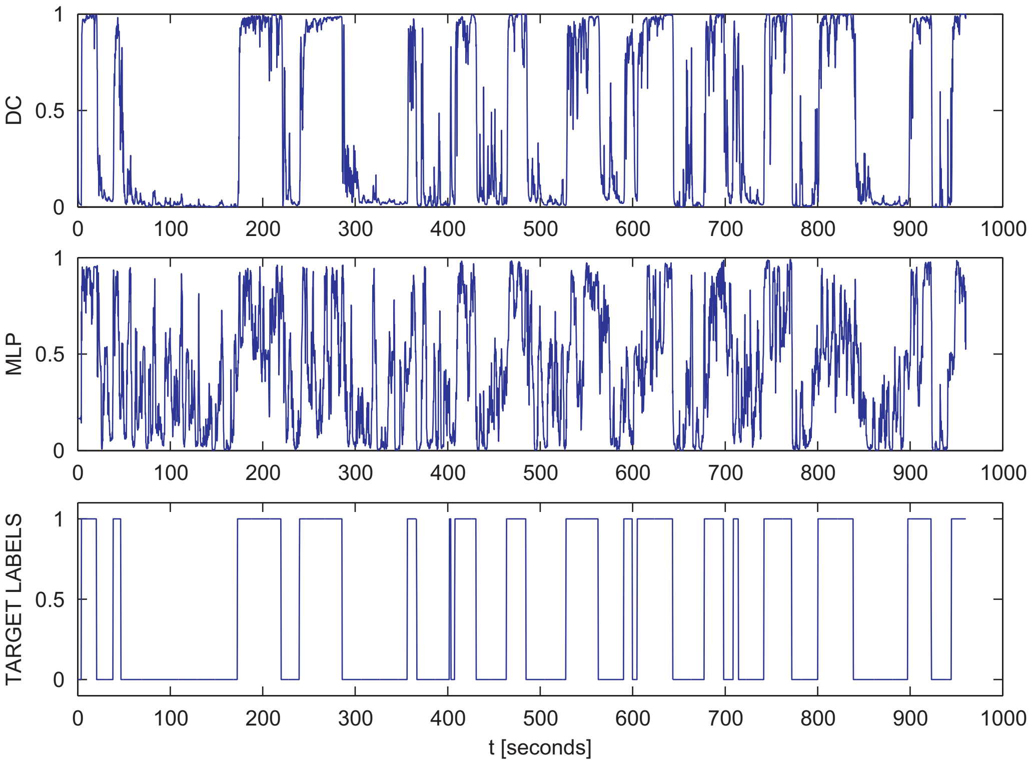

Fig. 6. Comparison of a section of output posterior from the dynamic model (DC) on the top plot, the MLP in the middle plot along with the true target labels (obtained retrospectively) shown in the lower plot. The *x*-axis represents time in seconds.

Table 2
Active label requesting: accuracy of the dynamic basis function model (DC), the static basis function model (SC) and the multi-layer perceptron (MLP) over eight experimental runs.

Method	expt. 1	expt. 2	expt. 3	expt. 4	expt. 5	expt. 6	expt. 7	expt. 8
DC	0.896	0.839	0.876	0.950	0.894	0.947	0.798	0.812
SC	0.611	0.623	0.722	0.914	0.823	0.939	0.555	0.675
MLP	0.616	0.561	0.636	0.920	0.574	0.880	0.542	0.631
% Labels	22.0	16.1	18.0	3.1	7.4	0.8	13.7	13.7

The DC was significantly better in all experiments ($p < 0.01$ using a Wilcoxon rank test).

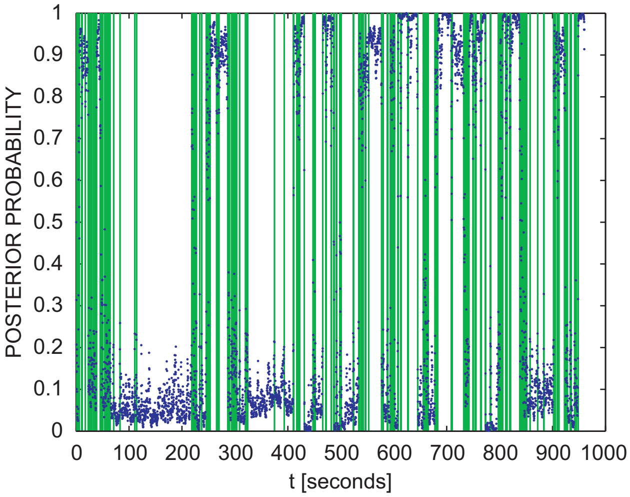

Fig. 7. BCI data: label requesting: a section of representative BCI data showing the posterior class probability on the *y*-axis along with label requesting (shaded regions). The *x*-axis represents time in seconds.

approach makes no explicit assumptions regarding the nature of the input feature encoding, merely that a discriminant boundary may be found which minimizes the fraction of incorrect decisions. The method has been applied in this paper to continuous numerical data but also works well with binary input strings. Care must be taken with nominal data (categorical values which are order independent), but the standard approach of converting a set of nominal values, lying between 1 and *K* say, into a corresponding *one-of-K* binary coding may be utilized. This algorithm, or any discriminant classification approach, may then be employed in the binary space and discriminant boundaries inferred.

6. Conclusion

Initial experiments have shown that this adaptive, non-linear method of classification is well suited to the non-stationary nature of decision making in an example biomedical problem. Moreover, we have shown that lack of labeled feedback is not incongruent to adaptive classification. Further study will include evaluation of the algorithm on a wider range of problem domains in an online environment.

Matlab code and sample data for this algorithm is available from: http://www.robots.ox.ac.uk/∼sjrob/Outgoing/dlr.html.

Acknowledgments

The BCI data used in these experiments were kindly provided by Dr. John Q. Gan and colleagues at the University of Essex. Work on this project is funded by the UK EPSRC. The authors would like to thank Jiwon Yoon, John Gan, Matt Dyson, Iead Rezek, Steve Reece, and Michael Osborne for discussions about this work.

References

[1] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.

[2] J.M. Bernardo, A.F.M. Smith, Bayesian Theory, Wiley, New York, 1994.

[3] T. Hastie, R. Tibshirani, Discriminant adaptive nearest neighbor classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (6) (1996) 607–616.

[4] W. Penny, S. Roberts, Dynamic logistic regression, in: Proceedings of IJCNN-99, 1999.

[5] M. Niranjan, Sequential Bayesian computation of logistic regression models, in: Proceedings of Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings, vol. 2, March 1999, pp. 1065–1068.

[6] P. Sykacek, S. Roberts, M. Stokes, Adaptive BCI based on variational Bayesian Kalman filtering: an empirical evaluation, IEEE Transactions on Biomedical Engineering 51 (5) (2004) 719–729.

[7] D.J.C. MacKay, A practical Bayesian framework for backpropagation networks, Neural Computation 4 (1992) 448–472.

[8] S. Roweis, Z. Ghahramani, A unifying review of linear Gaussian models, Neural Computation 11 (2) (1999) 305–345.

[9] S. Haykin, Kalman Filtering and Neural Networks, Wiley, New York, 2001.

[10] R.H. Shumway, D.S. Stoffer, Time Series Analysis and its Applications, Springer, Berlin, 2000.

[11] T.M. Vaughan, Brain computer interface technology: a review of the second international meeting, IEEE Transactions on Rehabilitation Engineering 11 (2) (2003) 94–109.

[12] P. Shenoy, M. Kraudelat, B. Blankertz, R. Rao, K-R. Müller, Towards adaptive classification for BCI, Journal of Neural Engineering 3 (2006) 13–23.

[13] P.L. Nunez, Electric Fields of the Brain, Oxford University Press, Oxford, New, York, 1981.

[14] J. Pardey, S. Roberts, L. Tarassenko, A review of parametric modelling techniques for EEG analysis, Medical Engineering and Physics 18 (1) (1996) 2–11.

[15] C. Andrieu, N. de Freitas, A. Doucet, Sequential MCMC for Bayesian model selection, in: IEEE Signal Processing Workshop on Higher Order Statistics, Ceasarea, Israel, June 14–16, 1999.

[16] E.A. Wan, R. van der Merwe, The unscented Kalman filter for nonlinear estimation, in: Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC), Lake Louise, Alberta, Canada, IEEE, 2000.

[17] J.-S. Lee, I.-S. Oh, Binary classification trees for multi-class classification problems, in: ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition, Edinburgh, Scotland, 2003, IEEE Computer Society, Silver Spring, MD, pp. 770–774.

About the Author—DUNCAN LOWNE graduated in computer science from Case Western Reserve University in 2004. After working in commercial biomedical signal analysis he started a PhD degree in the Pattern Analysis and Machine Learning Research Group in 2005. He tragically passed away in 2007. The methodologies he helped develop in this paper are witness to a life cut far too short. He is sorely missed.

About the Author—STEPHEN ROBERTS is Professor in Information Engineering. He has particular interests in the development of machine learning theory for problems in time series analysis and decision theory. Current research applies Bayesian statistics, graphical models and information theory to diverse problem domains including mathematical biology, finance and sensor fusion. He heads the Pattern Analysis and Machine Learning Research Group is a fellow of Somerville College and member of the Oxford-Man Institute for Quantitative Finance.

About the Author—ROMAN GARNETT received the A.B. degree in mathematics and the M.Sc. degree in computer science from Washington University in Saint Louis in 2004. He is currently working toward the DPhil degree in Engineering Science with the Pattern Analysis and Machine Learning Group at the University of Oxford. His research interests include data stream mining, data fusion, Bayesian learning, image and video processing, and reinforcement learning.