ON-LINE PROBABILISTIC CLASSIFICATION WITH PARTICLE FILTERS

Pedro A.d.F.R Højen-Sørensen[†] Nando de Freitas[‡] Torben Fog[§]

[†] Dept. of Mathematical Modelling
Technical University of Denmark
DK-2800 Kongens Lyngby, Denmark
phs@imm.dtu.dk

[‡]UC Berkeley Computer Science Division
387 Soda Hall, Berkeley
CA 94720-1776 USA
jfgf@cs.berkeley.edu

§ Research & Development, MAN B&W Diesel A/S
Teglholmsgade 41, DK-2450, Copenhagen SV, Denmark.
tof@manbw.dk

Abstract. In this paper, we apply particle filters to the problem of on-line classification with possibly overlapping classes. This allows us to compute the probabilities of class membership as the classes evolve. Although we adopt neural network classifiers, the work can be extended to any other parametric classification scheme. We demonstrate our methodology on a simple example and on the problem of fault detection of dynamical operated marine diesel engines.

INTRODUCTION

Sequential classification problems arise in a few areas of technology, including condition monitoring and real-time decision systems [13, 14]. For example, when monitoring patients, we might wish to decide whether they require an increase in drug intake at several intervals in time. Here, it is shown that particle filters [5] provide an efficient and elegant probabilistic solution to this problem. In particular, it becomes possible to compute the probabilities of class membership when the classes overlap and evolve with time. This classification framework applies to any type of classifier, but for demonstration purposes we focus on multi-layer perceptrons (MLPs).

We demonstrate the methodology on a fault detection problem. This application is of great importance as early detection of incipient faults can improve safety and efficiency, as well as, help to reduce down-time by automating planned maintenance in many industrial and transportation environments. Recently, it has been shown that batch trained neural network classifiers can be applied successfully to this problem [6]. However, in this paper we are considering fault detection of engines in dynamical use.

MODEL SPECIFICATION

We adopt the following Markov, nonlinear, state space representation to model the mapping between the inputs and outputs of the classifier

$$\text{Transition model: } p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}) \tag{1}$$

$$\text{Observation model: } p\left(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t\right) \tag{2}$$

where $\mathbf{x}_t \in \mathbb{R}^{n_x}$ denotes the input data at time t, $\mathbf{y}_t \in \{0,1\}^{n_y}$ represents the output class labels distributed according to $p\left(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t\right)$ and $\boldsymbol{\theta}_t \in \mathbb{R}^{n_\theta}$ corresponds to the parameters (weights) of a neural network $\mathbf{f}(\mathbf{x}_t, \boldsymbol{\theta}_t)$. (Note that the method also applies to other parametric classification schemes such as discriminant methods and on-line hidden Markov decision trees [10].) The parameters are assumed to follow a random walk $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mathbf{u}_t$. The process noise could, for instance, be Gaussian $\mathbf{u}_t \sim \mathcal{N}(\mathbf{0}, \delta_t^2 \mathbf{I}_{n_\theta})$, where $\mathbf{I}_{n_\theta}$ denotes the identity matrix of size $n_\theta \times n_\theta$. Other noise models for the evolution of the parameters are also possible. To complete the specification of the model, the initial state is $\boldsymbol{\theta}_0 \sim \mathcal{N}(\mathbf{0}, \delta_0^2 \mathbf{I}_{n_\theta})$. Note that the approach discussed in this paper can be used to model noise in the input $\mathbf{x}_t$. Moreover, it applies to time varying transition and observation densities.

For binary classification, the output data are assumed to be noiseless class labels $\{0,1\}$. We want the network output $\mathbf{f}(\mathbf{x}_t, \boldsymbol{\theta}_t)$ to represent the probability of class membership $\Pr(1|\mathbf{x}_t)$ of class $\{1\}$. The posterior probability of class $\{0\}$ is then given by $1 - \mathbf{f}(\mathbf{x}_t, \boldsymbol{\theta}_t)$. Considering this, the likelihood of the observations should be given by the following binomial (Bernoulli) distribution

$$p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t) = f(\mathbf{x}_t, \boldsymbol{\theta}_t)^{y_t}\left(1 - f(\mathbf{x}_t, \boldsymbol{\theta}_t)\right)^{1-y_t} \tag{3}$$

If one assumes that the outputs of the hidden layer neurons are exponentially distributed, then it follows that the output probabilities of class membership are logistic functions of the hidden layer outputs [3]. We, therefore, use a logistic output basis function to perform binary classification. This classification scheme can be straightforwardly extended to more output classes by adopting softmax basis functions and multinomial likelihood distributions [3].

Estimation Objectives

Our goal will be to approximate the posterior distribution $p(\boldsymbol{\theta}_{0:t}|\mathbf{d}_{1:t})$ and one of its marginals, the filtering density $p(\boldsymbol{\theta}_t|\mathbf{d}_{1:t})$, where $\mathbf{d}_{1:t} = \{\mathbf{x}_{1:t}, \mathbf{y}_{1:t}\}$. By computing the filtering density recursively, we do not need to keep track of the complete history of the parameters. We can also augment the state space to approximate the joint posterior of the parameters and hyper-parameters. For example, for Gaussian noise, we might be interested in estimating the distribution $p(\boldsymbol{\theta}_{0:t}, \boldsymbol{\delta}_{0:t}^2|\mathbf{d}_{1:t})$.

PARTICLE FILTERING

In recent years, many researchers in the statistical and signal processing communities have, almost simultaneously, proposed several variations of particle filtering algorithms: see [5] for a comprehensive review. Figure 1 illustrates

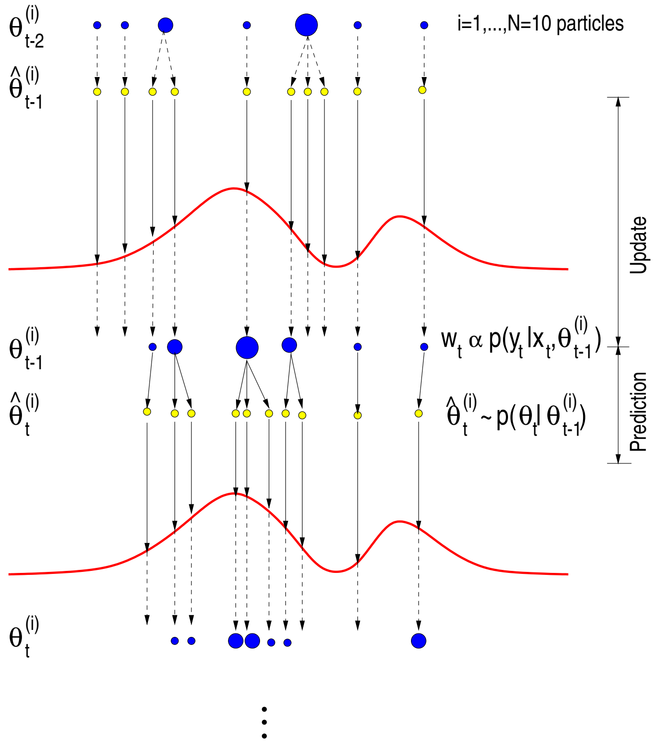

Figure 1: Update and prediction stages in a generic sequential Monte Carlo algorithm with the transition prior $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$ as proposal. At time $t-2$ the cloud of $N = 10$ samples provides an inadequate representation of the posterior. In the update stage, the likelihood $p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t)$ of each particle is evaluated. The size of the dark circles indicates the likelihood (importance weight w) of a particular particle. The particles are then selected according to their respective likelihoods. In this process, the particles with higher likelihood are allowed to have more "children". Subsequently, the algorithm computes the predicted values of the particles by passing them through the transition equations. The end result is that the surviving particles provide a better weighted description of the posterior distribution.

the operation of a generic particle filter. Only the fittest particles (samples), that is the ones with the highest likelihood in this case, are selected in an update stage (when the new data becomes available). These then proceed to be multiplied, according to their likelihood, in a prediction stage. In the next section, we shall present a generic particle filter for classification.

GENERIC PARTICLE FILTER FOR CLASSIFICATION

Given N particles (samples) $\{\boldsymbol{\theta}_{0:t-1}^{(i)}; i \in \{1, \ldots, N\}\}$ at time $t-1$, approximately distributed according to $p(\boldsymbol{\theta}_{0:t-1}|\mathbf{d}_{1:t-1})$, particle filters allow us to compute N particles $\boldsymbol{\theta}_{0:t}^{(i)}$, approximately distributed according to the posterior $p(\boldsymbol{\theta}_{0:t}|\mathbf{d}_{1:t})$, at time t. This is accomplished by sampling from the importance function $q(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{0:t-1}, \mathbf{d}_{1:t})$. We shall now present the general algorithm and, subsequently, discuss its main steps.

Generic Sequential Monte Carlo

1. Bayesian importance sampling step

   - For $i = 1, \ldots, N$, sample:

   $$\widehat{\boldsymbol{\theta}}_t^{(i)} \sim q(\boldsymbol{\theta}_{0:t}|\boldsymbol{\theta}_{0:t-1}^{(i)}, \mathbf{d}_{1:t})$$

   and set:

   $$\widehat{\boldsymbol{\theta}}_{0:t}^{(i)} \triangleq \left(\widehat{\boldsymbol{\theta}}_t^{(i)}, \widehat{\boldsymbol{\theta}}_{0:t-1}^{(i)}\right)$$

   - For $i = 1, \ldots, N$, evaluate the importance weights up to a normalising constant:

   $$w_t^{(i)} = \frac{p(\widehat{\boldsymbol{\theta}}_{0:t}^{(i)}|\mathbf{d}_{1:t})}{q(\widehat{\boldsymbol{\theta}}_t^{(i)}|\boldsymbol{\theta}_{0:t-1}^{(i)}, \mathbf{d}_{1:t})p(\widehat{\boldsymbol{\theta}}_{0:t-1}^{(i)}|\mathbf{d}_{1:t-1})}$$

   - For $i = 1, \ldots, N$, normalise the importance weights:

   $$\widetilde{w}_t^{(i)} = w_t^{(i)} \left[\sum_{j=1}^{N} w_t^{(j)}\right]^{-1}$$

2. Selection step

   - Multiply/Suppress samples $\widehat{\boldsymbol{\theta}}_{0:t}^{(i)}$ with high/low importance weights $\widetilde{w}_t^{(i)}$, respectively, to obtain N random samples $\widetilde{\boldsymbol{\theta}}_{0:t}^{(i)}$ approximately distributed according to $p(\widetilde{\boldsymbol{\theta}}_{0:t}^{(i)}|\mathbf{d}_{1:t})$.

3. MCMC step

   - Apply a Markov transition kernel with invariant distribution given by the product $\prod_{i=1}^{N} p(\boldsymbol{\theta}_{0:t}^{(i)}|\mathbf{d}_{1:t})$ to obtain $\boldsymbol{\theta}_{0:t}^{(i)}$.

■

Bayesian Importance Sampling Step

If we restrict ourselves to importance functions of the following form

$$q\left(\boldsymbol{\theta}_{0:t} \mid \mathbf{d}_{1:t}\right) = q\left(\boldsymbol{\theta}_0\right) \prod_{k=1}^{t} q\left(\boldsymbol{\theta}_k \mid \mathbf{d}_{1:k}, \boldsymbol{\theta}_{1:k-1}\right) \tag{4}$$

we can obtain recursive formulas to evaluate $w\left(\boldsymbol{\theta}_{0:t}\right) = w\left(\boldsymbol{\theta}_{0:t-1}\right) w_t$ and thus $\widetilde{w}_{1:t}$, w_t being given by

$$w_t \propto \frac{p\left(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{d}_{1:t-1}, \boldsymbol{\theta}_{0:t}\right) p\left(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_{t-1}\right)}{q\left(\boldsymbol{\theta}_t \mid \mathbf{d}_{1:t}, \boldsymbol{\theta}_{0:t-1}\right)} \tag{5}$$

There are infinitely many possible choices for $q\left(\boldsymbol{\theta}_{0:t} \mid \mathbf{d}_{1:t}\right)$, however we must make sure that its support includes the one of $p\left(\boldsymbol{\theta}_{0:t} \mid \mathbf{d}_{1:t}\right)$. For simplicity, we use the transition prior $p\left(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_{t-1}\right)$ as importance distribution for the MLPs. In this case, the importance weights are proportional to the likelihood: $w_t \propto p\left(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_t\right)$.

Selection Step

A selection (resampling) stage may be used to eliminate samples with low importance ratios and multiply samples with high importance ratios. A selection scheme associates to each particle $\boldsymbol{\theta}_{0:t}^{(i)}$ a number of "children", say $N_i \in \mathbb{N}$, such that $\sum_{i=1}^{N} N_i = N$. Several selection schemes have been proposed in the literature. These schemes satisfy $\mathbb{E}\left(N_i\right) = N \widetilde{w}_t^{(i)}$ but their performance varies in terms of the variance of the particles $var\left(N_i\right)$. Examples of these selection schemes include multinomial sampling [4], residual resampling [9, 12] and systematic sampling [11]. Their computational complexity is $\mathcal{O}\left(N\right)$.

MCMC Step

After the selection scheme at time t, we obtain N particles distributed marginally approximately according to $p(\boldsymbol{\theta}_{0:t} | \mathbf{d}_{1:t})$. Note that the discrete nature of the approximation can lead to a skewed importance weights distribution. That is, many particles have no children ($N_i = 0$), whereas others have a large number of children, the extreme case being $N_i = N$ for a particular value i. In this case, there is a severe depletion of samples. A strategy for improving the results involves introducing MCMC steps of invariant distribution $p(\boldsymbol{\theta}_{0:t} | \mathbf{d}_{1:t})$ on each particle [2, 7]. The basic idea is that, by applying a Markov transition kernel, the total variation of the current distribution with respect to the invariant distribution can only decrease. Note, however, that we do not require this kernel to be ergodic. Convergence results for this type of algorithm are presented in [7]. In the case of MLPs, we can use a smoothing Metropolis-Hastings [8] step as follows.

Smoothing Metropolis-Hastings step

- Sample $v \sim \mathcal{U}_{[0,1]}$.
- Sample the proposal candidate $\boldsymbol{\theta}_t^{\star(i)} \sim p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}^{(i)})$
- If $v \leq \min\left\{1, \frac{p(\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta}_t^{\star(i)})}{p(\mathbf{y}_t|\mathbf{x}_t,\widetilde{\boldsymbol{\theta}}_t^{(i)})}\right\}$
  - then accept move: $\boldsymbol{\theta}_{0:t}^{(i)} = \left(\widetilde{\boldsymbol{\theta}}_{0:t-1}^{(i)}, \boldsymbol{\theta}_t^{\star(i)}\right)$
  - else reject move: $\boldsymbol{\theta}_{0:t}^{(i)} = \widetilde{\boldsymbol{\theta}}_{0:t}^{(i)}$

  End If.

A SIMPLE CLASSIFICATION EXAMPLE

For demonstration purposes, we first consider a synthetic problem, where the target classes change with time. Specifically, the data was generated from two two-dimensional, overlapping and time-varying Gaussian clusters as depicted in Figure 2. If we look at all the generated data (bottom right of Figure 2), we note that non-sequential classification strategies would fail in this problem.

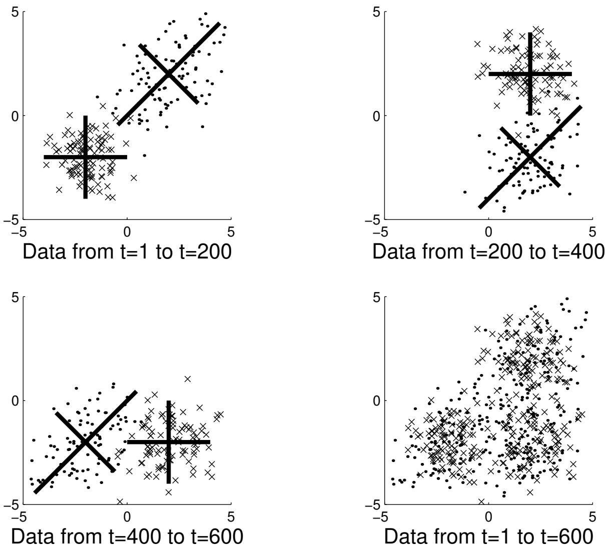

Figure 2: Data generated for the classification problem.

An MLP with 4 hidden logistic functions and an output logistic function was applied to classify the data. The network was trained, sequentially, with an

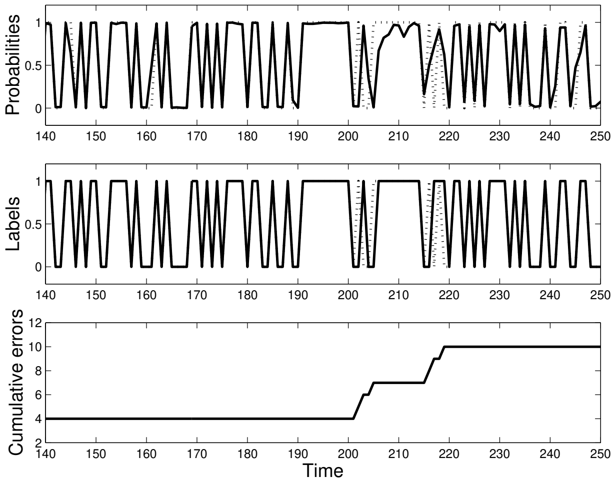

Figure 3: The top plot shows the true labels [$\cdots$] and one-step-ahead predicted probabilities of class membership [—]. The middle plot shows the predicted labels using a threshold of 0.5. The bottom plot shows the cumulative number of mis-classification errors.

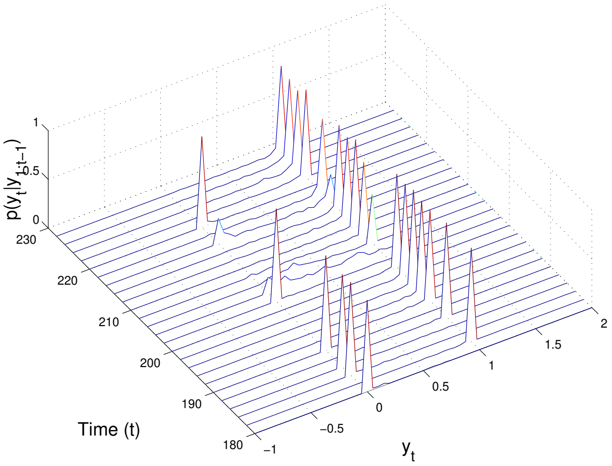

Figure 4: Probabilities of class membership for the classification problem.

SMC algorithm by proposing from the transition prior. A smoothing MCMC step was used to reduce the resampling variance. The number of particles was set to 200, the prior network weights were sampled from $\mathcal{N}(\mathbf{0}, 10\mathbf{I}_2)$, while their diffusion parameter was set to 0.2. Figure 3 shows the one-step-ahead predicted class probabilities, the output labels obtained by thresholding the output at 0.5 and the cumulative mis-classification errors. Figure 4 depicts the evolution of the probabilities of class membership. Despite the change in the distributions at $t = 200$ and $t = 400$, the algorithm recovers quickly.

AN APPLICATION TO FAULT DETECTION

In this section, we apply the proposed on-line classifier to monitor the exhaust valve condition in a marine diesel engine. The ability to detect valve burn-through, or leakage, in marine diesel engines is of great practical interest as it makes it possible to automate planned maintenance [6]. For instance, in the case of a minor leakage, it is possible to get the valve reconditioned and re-installed when the leakage is detected early. However, an undetected leakage will continue to aggravate the exhaust value condition (e.g. via hot corrosion) and hence affect the performance of the engine by reduction of pressure and power. If the leakage proceeds undetected, the damage will be too serious to allow for recondition. Thus, the main goal is to detect the leakage before the engine performance becomes unacceptable or irreversible damage occurs.

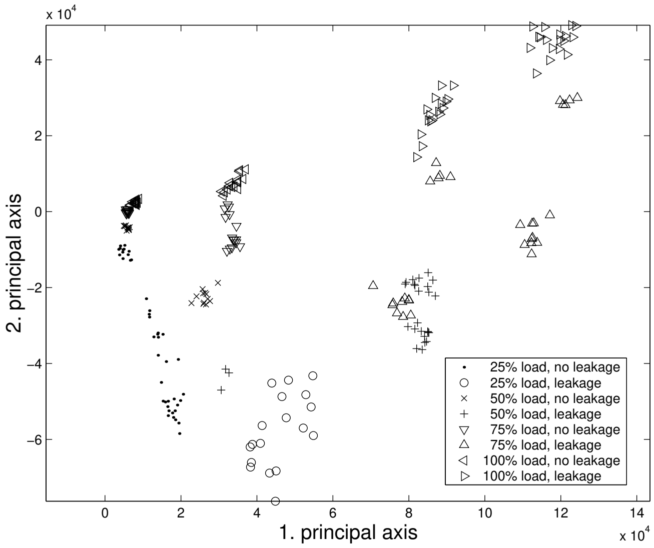

Figure 5: Shows the trigger sampled AE measurements projected onto the two first principal components. The two classes are approximately linearly separable

The non-intrusive monitored measurements consisted of vibrations and structure-borne stress waves also known as Acoustic Emission (AE) acquired during various engine load conditions (25%, 50%, 75% and 100%) and valve conditions (normal, small leakage and large leakage). The measurements were carried out using a 4 cylinder 500 mm bore marine diesel engine at a testbed where the propeller was simulated by a water brake. The RMS AE time-series was trigger resampled using a shaft timing signal obtained from an angle encoder pulse signal yielding 2048 angle positions per piston cycle. Additional dimensionality reduction of the input-space was obtained by using principal component analysis. In this paper, we we focused on classifying normal engine condition versus large valve leakage. Figure 5 shows the data projected onto the two first principal components. This figure suggest that

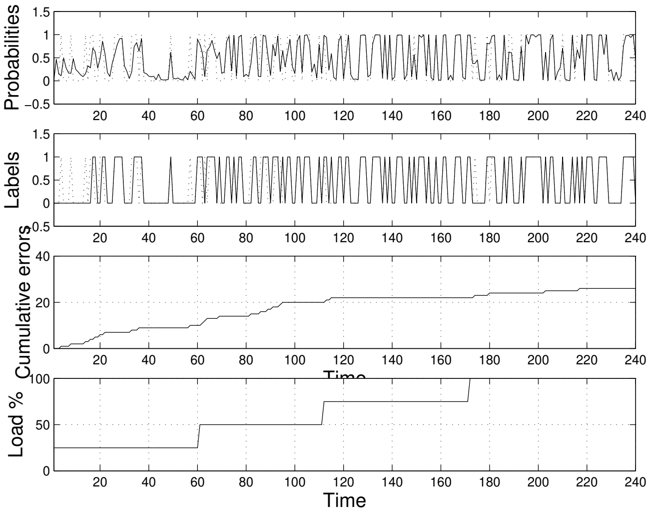

Figure 6: From top to bottom; The top plot shows the true label $[\cdots]$ and one-step-ahead predicted probabilities of class membership [—]. The next plot shows the the predicted classes using a threshold of 0.5. The next plot shows the cumulative number of mis-classification errors. The bottom plot shows the operating condition of the engine as a function of time

the two classes are approximately linearly separable. Due to the non-dynamic data acquisition, a temporal dynamic engine operation was simulated by collecting all data acquired with the same engine load and with the load level increasing in time (that is, 25%, 50%, 75% and 100%). Finally, for each fixed load level, the data was shuffled randomly.

An MLP with 2 hidden unit and 5 input nodes was used as to increase the modelling flexibility even though Figure 5 suggests that a linear discriminant on the data projected onto the two first principal components is sufficient to separate the two classes. Figure 6 shows the true label and one-step-ahead predicted probabilities of class membership obtained using 500 particles. As expected, we noticed that an initial convergence time was required for the classifier to start tracking the changes in the operating condition.

CONCLUSIONS

We presented a novel on-line classification scheme and demonstrated it on two problems. We believe this strategy has great potential and that it needs to be further tested on other types of parametric classifiers and classification domains. Some algorithmic improvements would result if one uses a discriminant strategy where the output neurons are linear and the measurement noise is Gaussian. In this case, it would become possible to obtain efficient Rao Blackwellised particle filters [1].

Acknowledgements

We are very thankful to Arnaud Doucet and David Lowe for their help and comments. The data was provided by MAN B&W Diesel's Research Center in Copenhagen, Denmark.

REFERENCES

[1] C. Andrieu, J. F. G. de Freitas and A. Doucet, "Sequential Bayesian Estimation and Model Selection Applied to Neural Networks," Techn. Report CUED/F-INFENG/TR 341, **Cambridge University Engineering Department**, May 1999.

[2] C. Andrieu, J. F. G. de Freitas and A. Doucet, "Sequential MCMC for Bayesian Model Selection," in **IEEE Higher Order Statistics Workshop**, Ceasarea, Israel, 1999, pp. 130–134.

[3] C. M. Bishop, **Neural Networks for Pattern Recognition**, Oxford: Clarendon Press, 1995.

[4] A. Doucet, "On Sequential Simulation-Based Methods for Bayesian Filtering," Techn. Report CUED/F-INFENG/TR 310, **Department of Engineering, Cambridge University**, 1998.

[5] A. Doucet, J. F. G. de Freitas and N. J. Gordon, **Sequential Monte Carlo Methods in Practice**, Springer-Verlag, 2000.

[6] T. L. Fog, L. K. Hansen, J. Larsen, H. S. Hansen, L. B. Madsen, P. Sørensen, E. Hansen and P. S. Pedersen, "On Condition Monitoring of Exhaust Valves in Marine Diesel Engines," in **Proceedings of the IEEE Workshop on Neural Networks for Signal Processing IX, Piscataway, New Jersey**, 1999, pp. 554–563.

[7] W. R. Gilks and C. Berzuini, "Monte Carlo Inference for Dynamic Bayesian Models," Unpublished. Medical Research Council, Cambridge, UK.

[8] W. K. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and their Applications," **Biometrika**, vol. 57, pp. 97–109, 1970.

[9] T. Higuchi, "Monte Carlo Filter Using the Genetic Algorithm Operators," **Journal of Statistical Computation and Simulation**, vol. 59, no. 1, pp. 1–23, 1997.

[10] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola and L. K. Saul, "An introduction to variational methods for graphical models," **Machine Learning**, vol. 37, pp. 183–233, 1999.

[11] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," **Journal of Computational and Graphical Statistics**, vol. 5, pp. 1–25, 1996.

[12] J. S. Liu and R. Chen, "Sequential Monte Carlo Methods for Dynamic Systems," **Journal of the American Statistical Association**, vol. 93, pp. 1032–1044, 1998.

[13] D. G. Melvin, "A Comparison of Statistical and Connectionist Techniques for Liver Transplant Monitoring," Techn. Report CUED/F-INFENG/TR 282, **Cambridge University Engineering Department**, 1996.

[14] W. D. Penny, S. J. Roberts, E. Curran and M. Stokes, "EEG-Based Communication: a Pattern Recognition Approach," To appear in *IEEE Transactions on Rehabilitation Engineering.*