Smart Feature Detection Using an Invariance Network Architecture

A.J.Lacey, N.A.Thacker and N.L.Seed
Dept. of Electronic and Electrical Engineering
University of Sheffield
Mappin St., Sheffield, S1 3JD
United Kingdom
email: a.lacey@sheffield.ac.uk

Abstract

The use of image features greatly improves the computational efficiency of subsequent machine vision tasks by directing processing effort to information rich areas of the image. However, the extraction of image features is an intensive process where the usual mapping between image data and feature is only approximately defined. Further, feature enhanced images provide very little information regarding the reliability of the information they embody. This results from the application of heuristics during the later stage of feature extraction.

This paper describes an artificial neural network trained to perform feature detection. The network is 'tuned' to extract a particular feature type. This is achieved using the functional mapping of an existing detection algorithm once systematic errors in this technique were removed. The scale of the mapping problem is reduced by enhancing the invariant characteristics of the feature. Only then is a manageable sized network able to perform the mapping task. The network enables the conditional probability of a corner to be used in the final image segmentation.

1 Introduction

Edges and corners represent the most commonly used types of image features. This is because both edges and corners have diffeomorphic equivalence to real 3D features. That is as differential features of the image they correspond directly to equivalent features in the world. Edges only constrain the location of the feature in one dimension a restriction which is more commonly known as the aperture problem [1]. The corner feature, however provides an unambiguous 2D localisation of a point/line object feature in an image.

Algorithms reported to detect corners can broadly be divided into two groups. The first group require the prior extraction of an edge string which is searched for points of high curvature or extrapolated to find line intersections on polygonal models [2]. The larger body of work centres on grey level image analysis and

BMVC 1995 doi:10.5244/C.9.33

the measurement of local gradients and surface curvature. A particularly good example of this work is that of Harris and Stephens [3], essentially a continuous, circular auto-correlation operation. However this technique is computationally expensive and requires heuristically selected parameters.

For any detection algorithm the definition of a 'corner' is unlikely to match the desired measure of a 3D feature exactly. The fundamental techniques used are often derived from convenient methods of image calculus and not an exact specification of the problem. This leads to systematic effects which introduce inaccuracies into the detection process. It also renders algorithms unreliable as such definitions take no account of typical errors in the image data. Therefore there is a need for a detection method which can reliably and accurately recover the location of image corners.

It has long been known that Artificial Neural Networks (ANN) are useful for functional mapping problems of the one to one or many to one variety [8]. ANNs are taught this functional mapping by example and therefore the desired input to output relationship can be enforced. Further, the ANN can be trained to present an output which approximates to Bayes conditional probabilities, see appendix A. Thus the ANN has the flexibility to adapt to a variety of corner definitions and facilitates the use of principled techniques in establishing corner locations.

An ANN has been trained to extract specific corners which lie at the intersection of approximately orthogonal edges in calibration grid images. Location probabilities derived from a corrected Harris and Stephens detector were used as target training outputs. A series of transformations were performed on the image to reduce the scale of the mapping problem. A MLP was trained and its' detection performance on unseen data compared to the known location of the vertices. By training the network on real image data it can be made reliable in the presence of image noise. The trained network estimates the probability that the Harris and Stephens detector would have located a corner at each point in the image.

2 The Probabilistic Response Map

The supervised training of a MLP network requires the prior definition of the desired output responses for given input patterns. In this case, a probability map which represents the likelihood that a corner can be found at the centre of the corresponding image pixel.

2.1 The Harris and Stephens detector

The corner detector described by Harris and Stephens is a reformulation of the Moravec operator [4]. In this model a corner is defined as a peak in the local auto-correlation patch. Where the Moravec operator is anisotropic, only discrete shifts of 45 degrees are considered, the Harris and Stephens model is continuous. Furthermore, using a circular Gaussian window improves the detector's noise characteristic and defines the scale of the feature to be detected.

However, the use of finite sized auto-correlation window introduces a bias between the detected and actual location of corners. This systematic shift was described by Wang, [5], who also showed that the error scaled in proportion to the

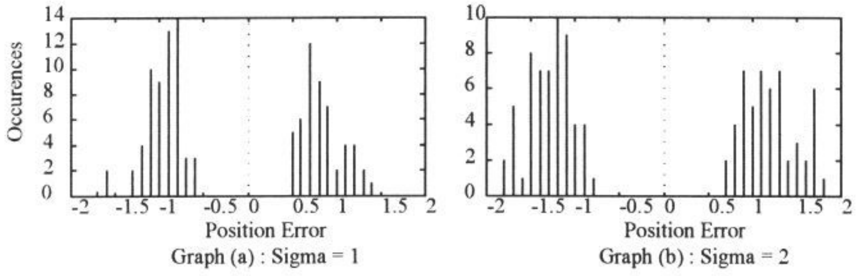


Figure 1: Corner Location Error Distributions

variance of the Gaussian window employed. It is possible to demonstrate this by comparison of detected corner locations to intersecting edges, using a grid image similar to figure 7(a). In this image there is an equivalence between the corners and the intersections of the edges. These edges have been detected using a Canny edge detector [6] with a known location accuracy. The distribution of errors between the two is shown in figure 1(a) for a corner detection with a variance scale of one pixel. Repeating the detection with twice the variance yields the distribution of figure 1(b) where the mean of the distributions has doubled in proportion.

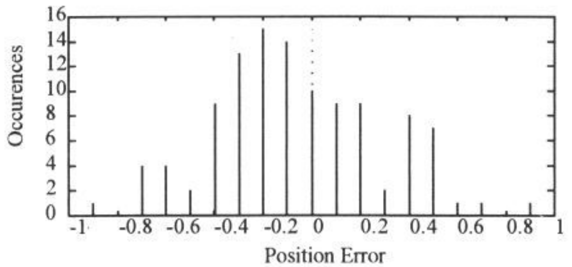


Figure 2: Corner Location Errors After Projection

Having quantified this error, it is a simple procedure to re-project the corners to their correct location (the theoretical location which an auto-correlation window of zero size would have found) as shown in figure 2. Although this process corrects the shift it results in a approximately 25% increase in the required computation. Also, as the variance in figure 2 shows, there remain inaccuracies in the detection produced by genuine error in the input data.

The probabilistic response map is built by blurring entries in the detected corner location image with this variance. The required response at each pixel location is then recovered by integrating the resulting probability density function over the active pixel region.

3 Training Data

Corners, being spatial features require support from a local patch of the image. Therefore a 7x7 region of pixels, centred on the proposed corner pixel, is used as input to the network.

3.1 Dimensional reduction

Standard feed-forward neural networks scale in approximately N^3 with the complexity of the mapping problem. Therefore the ANN should not be expected to extract the desired invariance properties from the data. Instead this information is made explicit by various transformations of the image data.

After taking the logarithm of the image and Gaussian blurring, the sum-squared-gradient of the image is computed in order to enhance discontinuous regions. We have already stated the diffeomorphic equivalence of corners to 3D features in the world. Therefore given it is these features we wish to extract, we lose no information by applying gradient operations to the image. Gradient information is used further, to align the principle image patch gradients with the x and y axis. This reduces the possible orientations that corners may take and therefore reduces the scale of the mapping problem. Without these processing stages the network would be required to learn all possible orientations and scales of corners vastly increasing the size of the required architecture.

These stages are summarised by the diagram of figure 3.

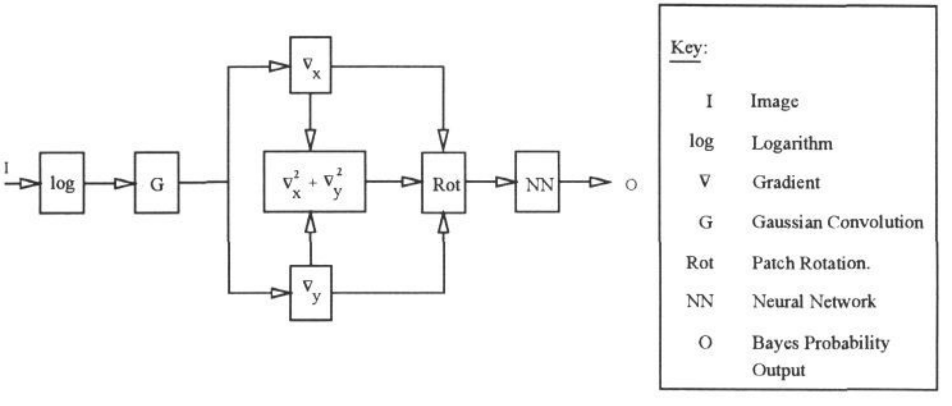


Figure 3: Processing Stages

3.2 Data-set reduction

Before being submitted to the training set the image data is 'thinned' or 'clustered' to produce a more manageable, unbiased set of training examples. A leader clustering algorithm is used to reduce the image data down to the minimum set of examples needed to provide complete coverage of the pattern space. However, the clustering process affects the corner to noncorner training ratio and thus the effective *a priori* probabilities. If the network is going to estimate the Bayes conditional probability is it not the case that this information must be preserved? For Bayesian classification the important result is;

$$\frac{P(corner|I_n)}{P(noncorner|I_n)} = \frac{P(I_n|corner)P(corner)}{P(I_n|noncorner)P(noncorner)} \tag{1}$$

where I_n is the image patch and $P(corner)$ and $P(noncorner)$ are the corner and noncorner *a priori* probabilities respectively. If the effective *a priori* probabilities are modified through clustering by a factor α;

$$\frac{P^*(corner|I_n)}{P^*(noncorner|I_n)} = \gamma \frac{P(corner|I_n)}{P(noncorner|I_n)}, \ \gamma = \frac{\alpha}{1-\alpha} \tag{2}$$

Therefore, provided at every point in the pattern space the ratio of corners to noncorners is preserved, modifying the *a priori* probability only scales the classification decision threshold.

It is also important that exact image characteristics are preserved and therefore the image data and probabilistic response are not averaged at cluster points. This is important if the ANN is to correctly interpolate the conditional probabilities of corners once it is trained.

4 Neural Classifier

The conventional MLP architecture, [7] is employed as the neural classifier, figure 4. The ability of this architecture to perform pattern classification can be attributed to its potential to create a specific nonlinear mapping of the input into a space spanned by the hidden nodes.

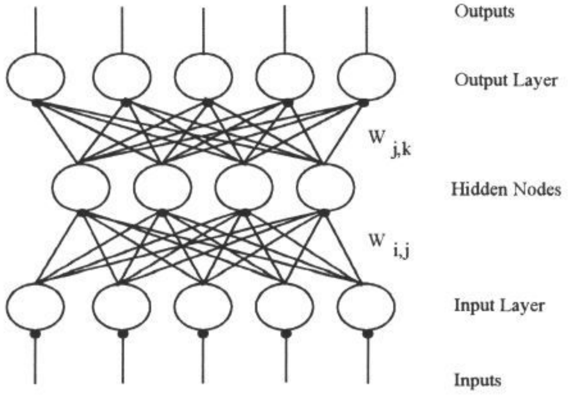


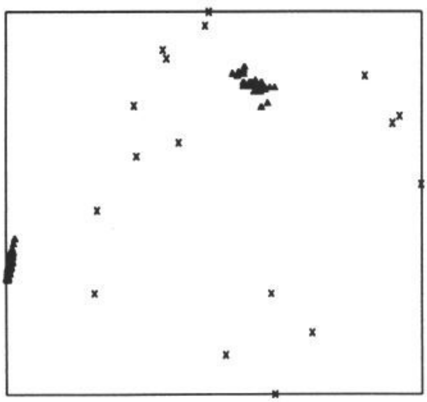
Figure 4: MLP Architecture

The traditional back-propagation algorithm has not been used to train this network as it requires the selection of several parameters for which optimal values cannot be calculated. Instead a conjugate gradient descent method was employed which effectively determines the value of any parameters at each step. The details of this technique are described in [9].

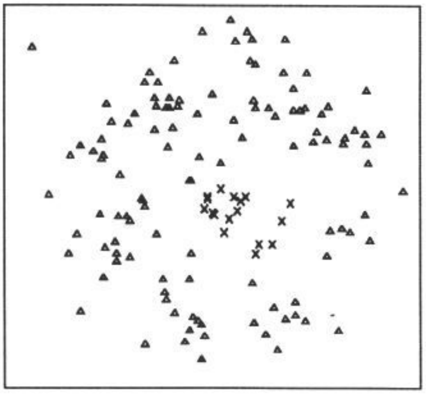
The number of hidden nodes required remains a free parameter. Without knowledge of exactly how the inputs are to map to the outputs a theoretical value for this cannot be calculated. Providing too few hidden nodes will inhibit the networks ability to find a useful functional mapping of the data. Too many hidden nodes and the network becomes unnecessarily large resulting in extended training times. In fact it has been suggested that using too many hidden nodes can actually have a detrimental effect on network performance, a problem known as the bias/variance dilemma, [10].

5 Data visualisation

By visualising the data we gain insight into the complexity of the classification problem. However the high-dimensionality of the data makes direct visualisation impossible. Instead, the mapping algorithm of Lee [11] was used to plot a subset of the Euclidean disparities between the image vectors, figure 5. Graph (a) shows the distribution of the image data without the preprocessing and graph (b) shows the data distribution after the preprocessing. The triangular nodes represent non-corner image vectors and the cross nodes represent 'corner-like' vectors, defined as those vectors with a target response greater than zero.



Graph (a)



Graph (b)

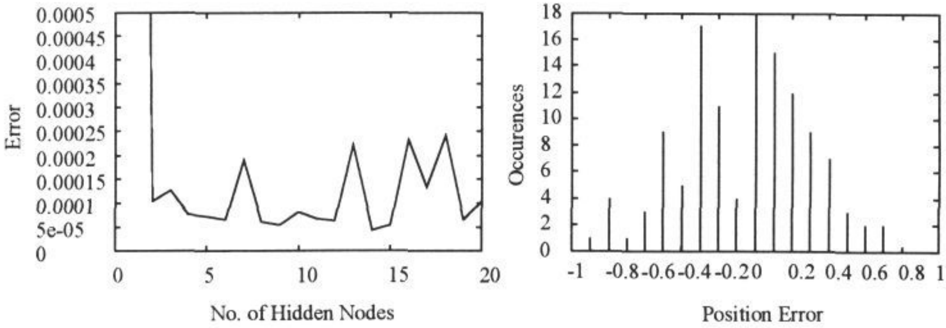
Figure 5: Euclidean Distance Image Maps

Image preprocessing has reduced the distribution of the corner nodes into a single cluster, graph (b). It has also affected the variation in the non-corner nodes producing a coherent cluster away from the corner nodes. The circular distribution of non-corner nodes in graph (b) is caused by a systematic effect of the mapping algorithm.

6 Performance

In order to produce a set of corners with a range of opening angles the calibration grid image was resampled with an aspect ratio of 0.5 these images were separated into halves. The upper half of the original image and the lower half of the affine projected image were then clustered to give 1779 image patch examples, of which 195 had a required response greater than zero.

A series of networks with increasing numbers of hidden nodes were trained on the clustered grid image data. Each trained network was applied to an image with 'intermediate type' corners; the affine warped 0.75 aspect ratio grid image of figure 7(a). The per-pixel error between the network response and the desired response for this image has been plotted, figure 6(a).

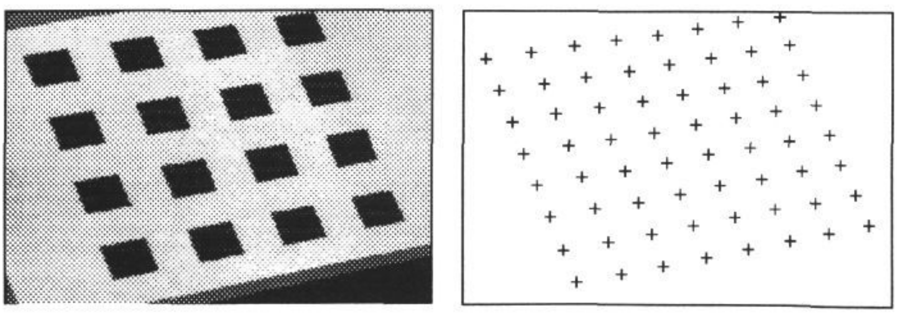


(a) Hidden Node Error

(b) 5 Hidden Nodes Error Distribution

Figure 6: Hidden Node Errors and Location Error Distribution

Using a network with 5 hidden nodes, corners in the warped grid image of figure 7(a) were located by fitting a quadratic function to local peaks in the network response map, figure 7(b), approximately reconstructing the original PDF. Comparison of these locations to intersecting Canny edges yielded the distribution of figure 6(b).

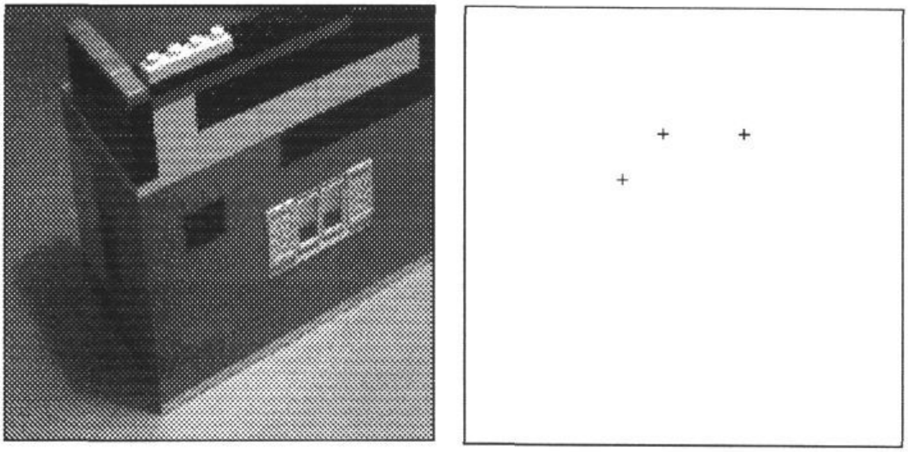


(a) 0.75 Aspect Ratio Grid Image

(b) Network Detected Corners

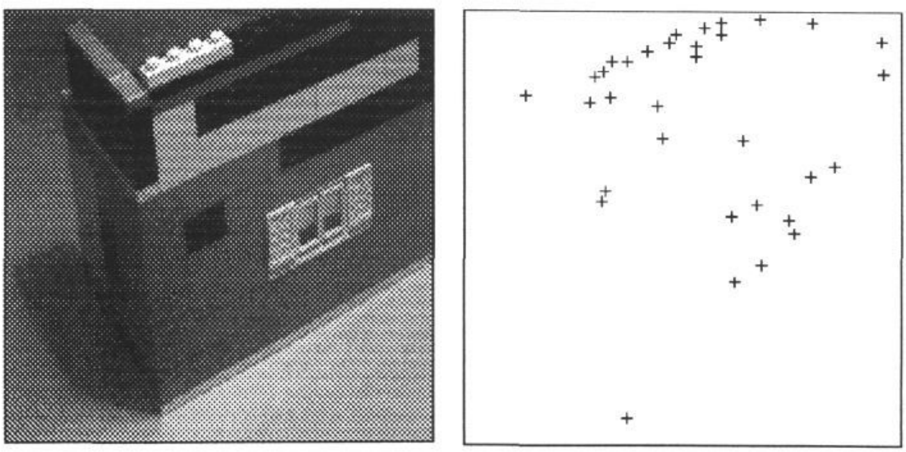
Figure 7: Intermediate Warped Grid Image and Corners

The 5 hidden node grid trained network was then used to detect corners in the right camera image from a stereo pair, figure 8(a). The only corners this network detects correspond to those similar to corners in the grid image, figure 8(b). A network with 8 hidden nodes was trained on data extracted from the left house image complementing figure 8(a). When this network was applied to figure 9(a) the corners of figure 9(b) were detected.



(a) Right House Image (b) Grid Trained Network Corners

Figure 8: Right House Image and Grid Network Corners



(a) Right House Image (b) House Trained Network Corners

Figure 9: Right House Image and House Trained Network Corners

7 Conclusions

A neural network feature detector is shown to have several benefits over conventional detection methods based on techniques of image calculus. The supervised training of the network enables arbitrary 'corner' definitions allowing the user to correct or re-define corner examples. This has been demonstrated by 'tuning' the network to one particular type of corner, that of the calibration grid image. The network only finds features of this type as demonstrated by figure 8(b). Such is the flexibility of this approach that a similar architecture has been trained to detect edges, [12].

The ability of neural networks to estimate conditional probabilities enables image segmentation based on corner probabilities as opposed to a geometric threshold.

Extracting the feature invariant properties of scale and orientation has a dramatic effect on the size of the network required when compared to the direct application of a feed-forward network. Assuming corners exist at 10 different grey-level scales and may occupy one of 10 discrete orientations we can expect a reduction in network size of the order of 100. A similar reduction can be expected in the size of the required training set and training time is reduced by several orders of magnitude.

In the absence of absolute grey-level information the network remains invariant to image noise. It is believed the network is using the image gradients to assess image smoothness, rejecting those patches with a broad distribution of gradients.

The computational requirements of the neural network detector are currently being analysed. The processing requirements of the algorithm are believed to be more appropriate to hardware implementation than existing methods. The algorithm can be formulated into a series of convolution processes which map well onto the Video Convolution Processor which has been designed within the group, [13].

Acknowledgements

We would like to acknowledge the useful advice of Dr Peter Rockett regarding this work. Dr Thacker is supported on SERC/ESPRC contract No. GR/J104464.

Appendix A: Estimation of Bayes Conditional Probabilities

If the network is trained using the least squares criteria as the optimisation function;

$$\epsilon_k = \sum_n (o(I_n) - t_{nk})^2$$

where $o(I_n)$ is the network output with application of input pattern I_n and t_{nk} is the desired output response. Provided the training data provides a *1-from-K* coding of the output, the error measure may be partitioned across the K classes according to their conditional probabilities $P(C_k|I_n)$, thus;

$$\epsilon_k = \sum_n \sum_k (o(I_n) - t_{nk})^2 P(C_k|I_n)$$

Leading to;

$$\epsilon_k = \sum_n (o(I_n) - E[t_n|I_n])^2 + \sum_n var(t_k|I_n)$$

This has separated the minimisation function into two components where the first part is known as the bias and the second part known as the variance. The variance is purely data dependent and therefore training can minimise only the bias. Clearly this is a minimum when $o(I_n) = E[t_n|I_n]$. In the limit of an infinite number of training samples $E[t_n|I_n] = P(C_k|I_n)$. Under these conditions a network trained with the least squares error function will approximate the conditional probability of classification. For an extended proof of this theorem the reader is directed to [14]

References

[1] S.Ullman., "The Interpretation of Visual Motion", MIT Academic Press, 1979.

[2] H.Asada and M.Brady., "The Curvature Primal Sketch", IEEE PAMI., pp 2-14, Vol 8, 1986.

[3] C.Harris and M.Stephens., "A Combined Corner and Edge Detector" Proceedings of the Fourth Alvey Vision Conference. pp.147-151 August 1988.

[4] H.Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover", Tech. Report, Carnegie-Mellon University, Robotics Institute, September 1980.

[5] H.Wang and M.Brady, "A Practical Solution to Corner Detection", Proceedings of IEEE International Conference on Image Processing, Vol. I, pp. 919-923, 1994.

[6] J.F.Canny, "A Computational Approach to Edge Detection", IEEE PAMI Vol 8, No. 6, 1986.

[7] D.E.Rumelhart, G.E.Hinton and R.J.Williams, "Learning Internal Representations by Error Backpropagation", in "Parallel Distributed Processing: Explorations in the Microstructure of Cognition", MIT Press, Cambridge, MA, pp 318-362, 1986.

[8] R.P.Lippman., "An Introduction to Computing with Neural Nets", IEEE ASSP Magazine, April 1987.

[9] N.A.Thacker, "Neural Networks in Machine Vision", Notes from 1995 ESPRC Machine Vision Summer School.

[10] S.Geman, E.Bienenstock and R.Doursat, "Neural Networks and the Bias/Variance Dilemma", Neural Computation, Vol. 4, No. 1, January 1992.

[11] R.C.T.Lee, J.R.Slagle and H.Blum, "A Triangulation Method for the Sequential Mapping of Points from N-Space to Two-Space", IEEE Trans. Comput., vol. 26, pp. 288-292, March 1977.

[12] W.C.Chen, N.A.Thacker and P.I.Rockett, "An Adaptive Step Edge Model for Self-consistent Training of a Neural Network for Probabilistic Edge Labeling", Submitted February 1995 to IEE Journal of Vision, Image and Signal Processing.

[13] R.A.Lane, N.A.Thacker, N.L.Seed and P.A.Ivey, "A Stereo Vision Processor", Proceedings of the IEEE CICC, May 1995.

[14] M.D.Richard and R.P.Lippmann, "Neural Network Classifiers Estimate Bayesian *a posteriori* Probabilities", Neural Computation, Vol. 3, pp 461-483, 1991