PERGAMON

Neural Networks 14 (2001) 1257–1264

Neural Networks

www.elsevier.com/locate/neunet

Contributed article

Novelty detection using products of simple experts—a potential architecture for embedded systems

Alan F. Murray*

Department of Electronics and Electrical Engineering, University of Edinburgh, Mayfield Road, Edinburgh, Scotland, EH9 3JL, UK

Received 1 November 2000; accepted 23 May 2001

Abstract

The 'Product of Experts' architecture (concentrating on binary-stochastic elements) is described in the context of its suitability for implementation as mixed-mode hardware, with algorithmic modifications that render the training procedure hardware-implementable. Results show that the PoE is capable of modelling non-linear, multi-dimensional data drawn from both artificial and real sources. The capability of the PoE to perform on-line novelty detection is described and demonstrated on both artificial and real data. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Probabilistic model; Novelty detection; Analogue VLSI; Generative model

1. Introduction

Neural algorithms and architectures have, to date, been either:

1. useful, well-researched and awkward to implement in compact analogue hardware (e.g. the MultiLayer Perceptron);
2. amenable to implementation by analogue hardware, but not demonstrably useful in solving practical, difficult problems (Hebbian Training (Hebb, 1949)).

From the current interest in probabilistic generative models has come the Product of Experts (PoE) (Hinton, 1999, 2000b). The generic PoE is not especially hardware-friendly, but the binary-stochastic PoE or Restricted Boltzmann Machine (Smolensky, 1986) is uniquely hardware-amenable, with extremely interesting properties. This paper demonstrates both of these qualities.

Section 2 presents an overview of the underlying PoE philosophy and approach with a detailed description (without derivation) of the Binary-Experts PoE and its training procedure. The algorithmic requirements are described along with an approach to implementation as mixed-mode VLSI. Section 2.2 also examines the binary PoE as a computational architecture amenable to noisy, small-geometry analogue VLSI and Section 3 presents simulation results of a binary PoE, trained on-line and acting as a data density model and 'Novelty Detector' for both artificial and real (medical) data.

2. Products of experts

The product of experts architecture is an example of a self-organising structure (Grossberg & Carpenter, 1991) that takes the form of a generative model (Mclachlan & Basford, 1988) that can model multi-dimensional data $\{\mathbf{d}\}$.

Additive mixture models suffer from a dichotomy between the need for broad elements ('experts' in Hinton, 1999) to cover high-dimensional data spaces and narrow experts to model sharp, multi-dimensional posterior distributes. The PoE replaces the additive, or disjunctive, mixture:

$$p(\mathbf{d}|\phi_1 \ldots \phi_N) = \sum_m C_m \times p_m(\mathbf{d}|\phi_m) \tag{1}$$

with a multiplicative, or conjunctive, distribution:

$$p(\mathbf{d}|\phi_1 \ldots \phi_N) = \frac{\Pi_m p_m(\mathbf{d}|\phi_m)}{Z} \tag{2}$$

where Z is the renormalizing factor,

$$Z = \sum_c \Pi_n p_n(\mathbf{c}|\phi_n) \tag{3}$$

Products of broad distributions can produce arbitrarily narrow (normalised) distributions. A PoE can be trained to

* Tel.: +44-131-650-5589; fax: +44-131-650-6554.
E-mail address: a.f.murray@ee.ed.ac.uk (A.F. Murray).

PII: S0893-6080(01)00097-1

model data $\{\mathbf{d}\}$ by adjusting the model parameters (the parameters in experts $\{\phi_m\}$) to follow the gradient of the log-likelihood of $\{\mathbf{d}\}$. Fitting the PoE to a set of data involves, ideally, following the derivative of the log-likelihood of the data $\mathbf{d}$, under the N-expert model, given by Eq. (4).

$$\frac{\partial \log p(\mathbf{d}|\phi_1 \ldots \phi_N)}{\partial \phi_m} = \frac{\partial \log p_m(\mathbf{d}|\phi_m)}{\partial \phi_m} - \sum_c p(\mathbf{c}|\phi_1 \ldots \phi_N) \frac{\partial \log p_m(\mathbf{c}|\phi_m)}{\partial \phi_m} \quad (4)$$

The second term in Eq. (4) represents the derivative of $\log(Z)$. The difficulty in estimating this term is that of generating 'fantasy' data with the correct distribution within the PoE model. One method of achieving an unbiased estimate is to use Gibbs sampling, allowing the process to converge to equilibrium. Generating correctly-distributed model data by full Gibbs sampling is, however, extremely computationally-intensive and a streamlined approach has been described, justified as 'one-step Gibbs sampling' (Hinton, 1999) or 'minimisation of contrastive divergence (Hinton, 2000b) that works well. The procedure is:

1. Calculate the posterior distribution across all of the experts in parallel (the details depend upon the nature of the individual experts) for each datum $\mathbf{d}$.
2. Use this posterior to select a value for each expert (latent variable).
3. Compute the distribution across the visible units *given* the states of the hidden units (experts).
4. Calculate the posterior across all of the experts in parallel for each datum $\hat{\mathbf{d}}$.

The training process follows the gradient of the *difference* between the divergences of the data vectors $\{\mathbf{d}\}$ and the one-step reconstructions $\{\hat{\mathbf{d}}\}$. This procedure has only been justified intuitively (Hinton, 2000b), but it works extremely well. The most persuasive intuitive arguments are that: (i) one step of Gibbs sampling is likely to produce a move in the right direction, if not of the correct magnitude and (ii) minimising the contrastive divergence makes regeneration of an actual datum more likely than that of data nearby, but not equal to the real datum. If the data lies on or near a smooth multi-dimensional manifold, this will tend to pull the model back towards it. We will not repeat the arguments in Hinton (1999, 2000b) in full, or to try to improve upon them. We will, instead, investigate their implications in a particular practical implementation on interesting data.

2.1. Binary experts—a restricted Boltzmann machine

If the experts comprise a single layer of *binary stochastic elements* obeying

$$p(S_j = 1) = \sigma\left(\sum_i T_{ji} S_i\right), \quad (5)$$

where $\sigma()$ is a sigmoidal function, the weight update that minimises contrastive divergence is (Hinton, 1999)

$$\Delta T_{ij} = \epsilon[<S_i S_j>^+ - <S_i S_j>^-] \quad (6)$$

where $<S_i S_j>^+$ is the expectation value of the product of the visible and hidden states, S_i and S_j, respectively, that result when data $\{\mathbf{d}\}$ is applied to the visible units, $d_i \rightarrow S_i$. Similarly, $<S_i S_j>^-$ is the same expectation value that results when one-step-reconstructed data $\{\hat{\mathbf{d}}\}$ is applied to the visible units, $\hat{d}_i \rightarrow S_i$. Crucially, the value of S_i that generates the one-step reconstruction $\hat{\mathbf{d}}$ is a (binary) sample from posterior distribution of the experts. Eq. (6) is extremely simple, requiring only information that is local to a connection (the states of the sending and receiving units, S_j and S_i). It also requires only two multiplications and one subtraction. Furthermore, for uncorrelated, zero-mean errors (noise) ϵ_i and ϵ_j on units i and j respectively:

$$<(S_i + \epsilon_i)(S_j + \epsilon_j)> = <S_i S_j> \quad (7)$$

so the learning process should be: (a) robust against some of the vagaries of hardware and (b) able to learn both on-line and in batch mode. It can also be further simplified for hardware purposes, with little loss of performance and no loss of principle. The Product of Experts has already been shown to be a powerful 'feature-detector' and classifier of handwritten digits, extracting features autonomously and creating diversity amongst the experts (Hinton, 2000b). In Hinton (2000b), performance is gauged by the perceived 'quality' of the extracted features and the ability to classify digits by using multiple PoEs, each trained on a different digit. Classification is achieved by choosing the model that maximises the log-likelihood of the data. In this paper, we look at a different use of the PoE model and thus a different performance metric. We show that the binary PoE does work in an interesting and useful manner on simulated and real data.

2.2. (Simple) products of experts and analogue VLSI

Eq. (6) is more amenable to analogue VLSI than is, for example, back-propagation. It is simple, does not require global communication of errors and is based explicitly upon the expectation values of simple two-component products. Expectation values are insensitive to the zero-mean noise that makes many other processes impractical in analogue VLSI. Studies have already been made of the effects of noise in various forms of neural network (Taylor & Bressloff, 1990; Murray & Edwards, 1993). Furthermore, reducing dimensions in analogue VLSI (device dimensions of 0.05 μm = 50 nm are now routine in research facilities) increase mechanisms such as 'flicker' and 'shot' noise. Such 'deep-sub-micron' devices still behave as transistors, but they add (significant) noise to any computation. While this

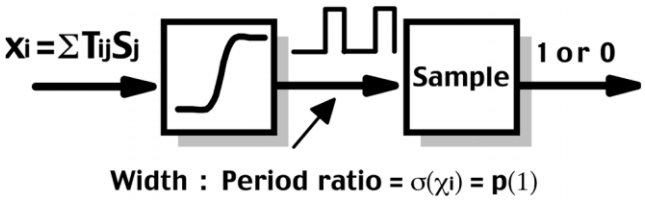

Fig. 1. Pulsed binary stochastic neuron (schematic).

is damaging to most conventional algorithms, neural or other, it is essential to the operation of stochastic elements. Analogue or mixed-mode ('pulsed') VLSI implementations of such architectures should therefore be explored in anticipation of the ever-smaller and noisier computational elements that are 50 nm MOS devices. The PoE is based on the same sum-of-products and sigmoidal activation function as is the MLP. In the pulse domain, therefore, the arithmetic elements described in Murray and Tarassenko (1994); Murray and Woodburn (1998) will suffice. The 'new' requirement is that of a binary stochastic neuron. We adopt an approach (Astaras, Dalzell, Murray & Reekie, 1999) that casts $p(S_j = 1) = \sigma(\Sigma_i T_j i S_i)$ as the (pulse-width):(pulse-period) ratio of an asynchronous pulsed waveform as illustrated in Fig. 1. When the waveform is sampled, the correct form of probabilistic behaviour results.

The PoE architecture and training process are therefore amenable to analogue hardware and thus to embedded systems that make use of the PoE's computational abilities. We are developing PoE hardware, but this paper aims to report what those abilities could be and how they could be useful. Further hardware results will be reported later.

2.3. Training procedure

The weight-update equation is admirably simple (Eq. (6)). As it uses expectation values, it should work in both 'batch' training mode and on-line (whereby weights are adjusted after each datum). On-line training has been used for all of the experiments on heartbeat data in Section 3.3 of this paper and works well. This bodes well for the PoE as an approach to modelling and 'tracking' real, time-varying data in real-time. It has also been suggested (Hinton, 2000a) that the use of binary-sampled values of the $\{S_i\}$ and $\{S_J\}$ everywhere in Eq. (6) should work, if more slowly. The latter approach can be shown to converge towards a solution, as suggested, but the coarse-quantisation of the $\{S_i\}$ prevents training from reaching a suitable minimum. In effect, $(<S_i^+ S_j^+> - <S_i^- S_j^->)$ never 'settles' and the final state of the training process is inherently extremely noisy.

We have, however, made a simplification to Eq. (6) for analogue hardware:

$$\Delta T_{ij} = \epsilon \,\mathrm{sign}[<S_i S_j>^+ - <S_i S_j>^-] \tag{8}$$

This simple modification means that weight changes are always 0 or $\pm\epsilon$, implemented by 'weight-bump' circuitry whose primary aim is to ensure that $+\epsilon$ does indeed mean 'up', $-\epsilon$ 'down' and that 0 really is 0. This distinction between a magnitude-correct weight change and this simpler, restricted-value update may sound trivial. In analogue design, it is not! It should, however, be noted that this modification may create training idiosyncrasies if combined with on-line (as opposed to batch) training. Small and large gradients of opposing sign, from individual training data, are constrained to have the same effect on a weight and thus an individual on-line training step may not move in the direction of steepest descent. Fig. 2 shows PoE training using Eqs. (6) and (8), on one of the artificial datasets described in Section 3.3. The PoE converges to a solution of equal 'quality' in both cases, but training times are longer when Eq. (8) is used.

3. Applications—data modelling

Our initial interest in probabilistic data modelling architectures centred on the Helmholtz Machine (Dayan, Hinton, Neal & Zemel, 1995; Dalzell & Murray, 1999), as a 'front-end' in sensor fusion systems. Such systems, particularly when integrated on silicon, as in an 'electronic nose', are notoriously noisy, difficult-to-calibrate and subject to sensor drift. This potential application is a suitable context within which to explore the potential of the PoE, using artificially generated, controllable, data.

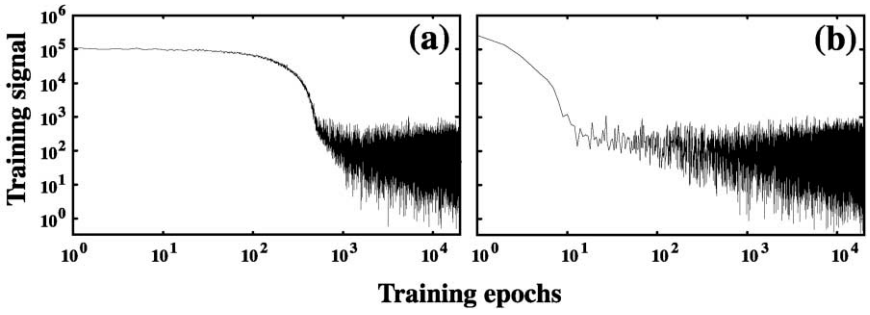

Fig. 2. (a) PoE training using Eq. (8) and (b) using the 'full' form of Eq. (6).

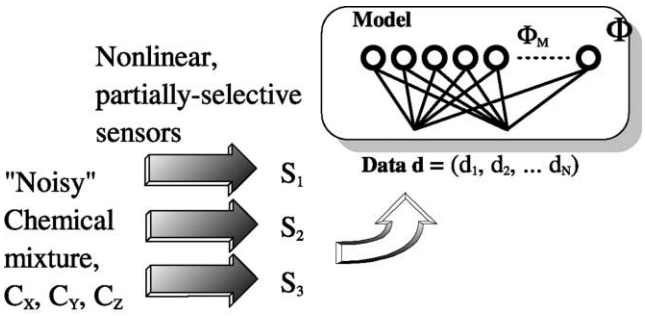

Fig. 3. Fictitious sensor system used for generating artificial data for simple PoE experiments.

3.1. Artificial data

The dataset emulates a three-sensor system, illustrated schematically in Fig. 3. A mixture of chemicals X, Y and Z with concentrations $\mathbf{C} = (C_X, C_Y, C_Z)$ is passed to sensors $S_1 \rightarrow S_3$ that are partially-selective and non-linear. The sensor outputs are given by:

$$\mathbf{S} = \mathbf{T}_1\mathbf{C} + \mathbf{T}_2\mathbf{C}^2. \tag{9}$$

$\mathbf{T}_1$ and $\mathbf{T}_2$ are mixing matrices of first and second order, respectively. $\mathbf{T}_1$ describes the linear response of the sensors (and would thus be diagonal for 100% selective sensors) and $\mathbf{T}_2$ describes the non-linear response—in this simple example, a term of the form $\mathbf{C}^2$. For the data below, all elements of the vector $\mathbf{C}$ are non-zero, the diagonal terms of $\mathbf{T}_1$ are the largest, and $\mathbf{T}_2$ is small but non-zero, and diagonal. In other words, the sensors are selective, but not 100% so, all inter-sensor mixing occurs via $\mathbf{T}_1$ and non-linear terms are small but non-zero. These details do not have great significance here, but they give even these artificial experiments a semi-realistic context. Fig. 4 shows projections of the generated three-dimensional data on to the three two-dimension planes, along with a two-dimensional plot of the activity of the first two experts in a single-layer PoE trained on this data.

Data was generated in three concentration clusters, before mixing and applying the non-linearities in Eq. (9). The three groups are discernible in the data projections, but are far clearer in the response of the experts, despite their apparent simplicity. Further confirmation that an acceptable generative model of the data has been built can be gained from Fig. 5, which shows a single two-dimensional projection of the data and its associated one-step (stochastic) reconstructions. The reconstructions are clearly 'credible' but not perfect. This measure of model quality is flawed, however, as a network that converges slowly under Gibbs sampling will always produce one-step reconstructions that are close to the training data. To eliminate the effect of this flaw in our experiments, therefore, we compared one-step reconstructions with the results of protracted Gibbs sampling in the model that produced Fig. 5. The distributions are gratifyingly similar, allowing some further confidence that the PoE model is, indeed, useful. One-step reconstruction is a stochastic process—giving rise to the differences in detail between the real and reconstructed data in Fig. 5 and to the 'noisiness' of the solution when training is essentially complete in Fig. 2. Fig. 4 suggests that the PoE has captured much of the structure in this data. The experiments in Section 3.2 use this PoE model to look for 'novelties' in the data and confirm that this simple, on-line trainable model is sufficiently good to be useful.

3.2. Novelty detection: artificial data

The principled way to detect novelty in this context would be to calculate the log-probability of a new data element, presented to a trained PoE, labelling as 'novel' all data that fall below a chosen threshold. The log-likelihood of a data vector $\mathbf{d}$ is given by (Hinton, 2000b)

$$L(\mathbf{d}) = \sum_{\mathbf{m}} [\log(p(\mathbf{d}|\phi_{\mathbf{m}}))] - \log(Z) \tag{10}$$

This measure was used in classification experiments (Hinton, 2000b), using multiple PoE models, trained on different data classes, as discriminators. This technique produces excellent results by assigning a class label to a new datum that corresponds to the PoE model assigning the highest value of $L(\mathbf{d})$. This is a difficult operation in analogue hardware. $\log(Z)$ is intrinsically difficult to estimate, but it is a constant and is irrelevant in the context of novelty detection, which involves relative likelihoods of 'normal' and 'abnormal' data. Calculating the $\{\log(p(\mathbf{d}|\phi_{\mathbf{m}}))\}$ involves substantially more complex analogue hardware, however, and we have examined a simpler,

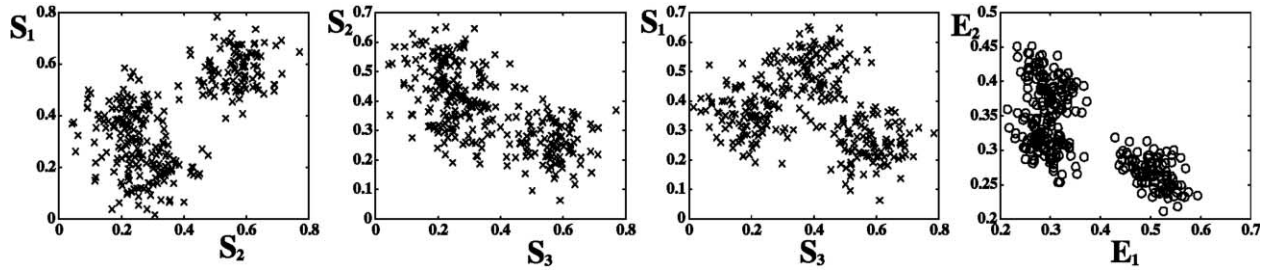

Fig. 4. Two-dimensional projections of three-dimensional data $S_1 \rightarrow S_3$, after mixing and non-linearity with the two-dimensional response of the first two experts, E_1 and E_2.

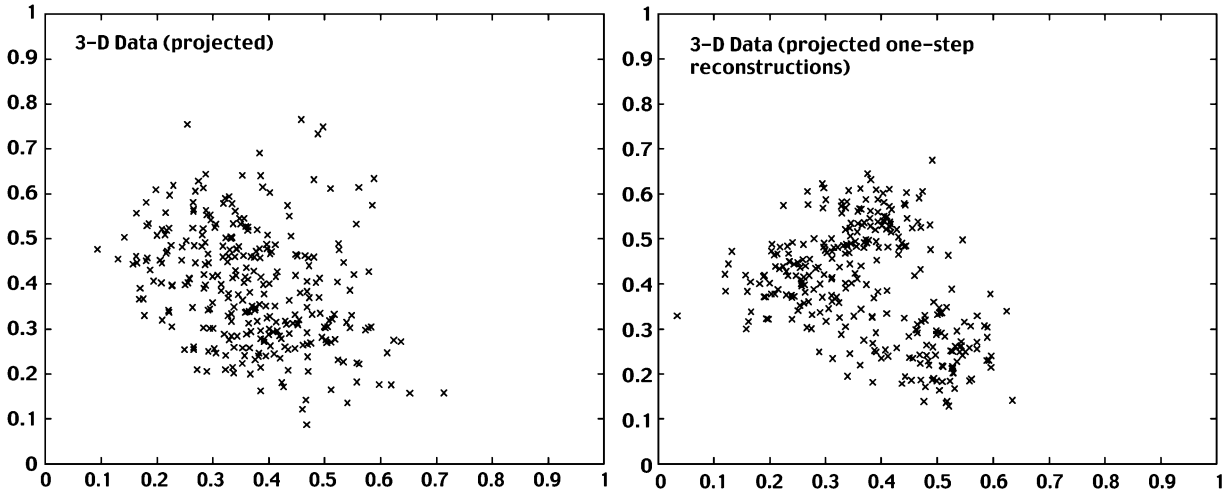

Fig. 5. Two-dimensional projections of three-dimensional, artificial, sensor data and 'one-step reconstructions' of the same data from a PoE.

less formally correct approach to novelty detection. In a well-trained PoE model, the average 'training signal' of Eq. (6) will be small for data that are well-modelled and large for data that are not. The magnitude of the training signal is calculated of necessity in every training cycle. It is therefore reasonable to study the following approach to novelty detection:

1. Train a PoE (on-line, in the interests of realism) to completion on a dataset, then set $\epsilon = 0$ in Eq. (6)
2. Continue to calculate the 'novelty signal'

$$N = \sum_{ij} (<S_i S_j>^+ - <S_i S_j>^-)^2 \tag{11}$$

3. Use N as a measure of data novelty for data outside the training set.

We will, however, compare this simpler novelty measure with the log-likelihood given in Eq. (10) to gauge the performance of the hardware-friendly form. On-line training is particularly interesting for an embedded system, where batching data may not be an option and tracking data as it evolves and changes may be desirable. Fig. 6 shows the results of such an experiment and several interesting features. The novelty signal is noisy, even once a solution has been reached, owing to the stochastic nature of the training process. However, the change to the data (the introduction of 'novelty') causes a marked change in the expectation value of N. The model can, however, retrain to take account of the new data. Intriguingly, but not surprisingly, given the nature of the cost function, the model is also sensitive to changes in the data *variance*. The mean of N changes with either a reduction or an increase in the data variance. The lower trace in Fig. 6 shows that changes to the data are also reflected in the negative Free Energy (or log-likelihood), confirming that the appearance of novelty as manifest in the simpler novelty signal N, is both correct and potentially usable.

Clearly, this simple model could, potentially, be used at least for novelty detection, using a hardware-amenable architecture and training process. Furthermore, the model's ability to re-adapt to changing data gives it an ability to track gradual data changes that do not signal novelty. We plan to add a second layer of experts (Hinton, 2000b) to capture the statistical structure in the activity of the experts (using greedy training, one layer at a time). We hope that the first layer will adapt to changing sensor gains, non-linearities, noise characteristics and mixing coefficients while the second will result in expert activations that can be used as inputs to a (simple) supervised classifier. At this stage, however, we will extend the encouraging results in Section 3.2 on artificial data, into real datasets.

3.3. Real data

The data in this initial study is that used by Tarassenko and Clifford (2001) to study the effectiveness of an MLP auto-associator for detection of abnormalities in ECG traces (in particular, ventricular ectopic beats, VEBs). This data is interesting because: (a) it is a real and important problem—detectors of regular heartbeats (QRS complexes) are triggered by VEBs and some form of novelty detection is desirable and (b) the problem is known to be soluble, but non-trivial (Tarassenko & Clifford, 2001). We followed the same event-detection strategy as described in Tarassenko and Clifford (2001), using a band-pass filter, differentiation and squaring to 'segment' and centre 1 s, 64 sample/s, QRS, VEB (or artifact) traces for presentation to a PoE with 64 (real-valued) inputs.

Several sizes of databases (number of heartbeats) have been used and differing numbers of experts. For the results shown in this paper, 300 s of data and six experts were used. Fig. 7 shows the results from a fully-trained PoE on both a regular heartbeat and a VEB.

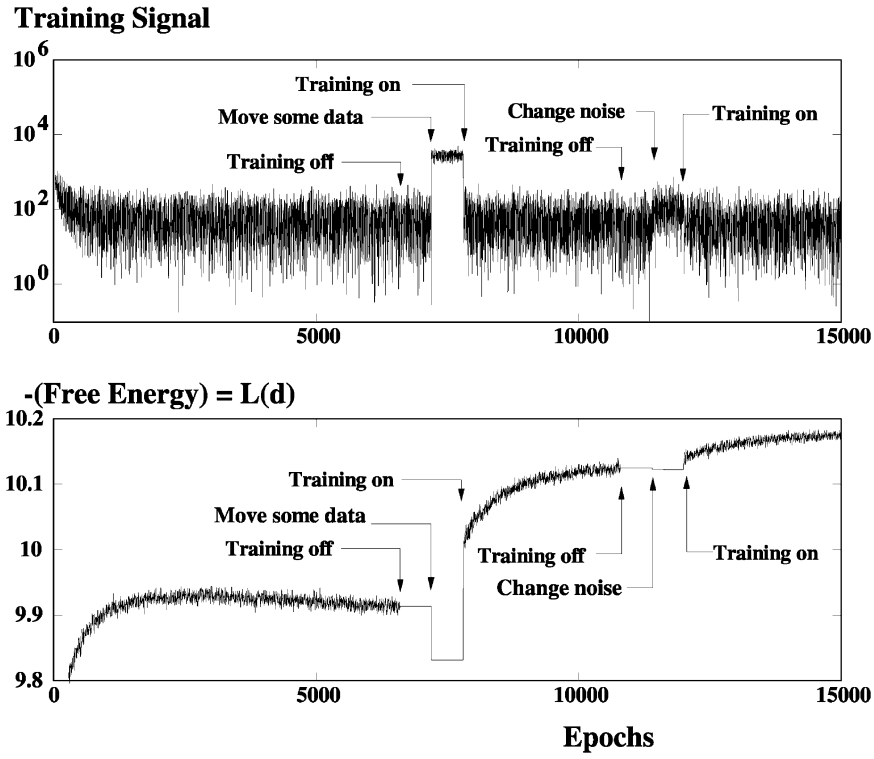

Fig. 6. The "Novelty Signal" N and the (average) log-likelihood of the data L (**d**) for a PoE under conditions of: (a) Initial training, switched off at epoch 6500 (b) A small change in the data at epoch 7200 (altering the co-ordinates of a cluster by 10%) (c) A short re-training period to adapt to the new data (epochs 7800 → 10800) (d) Altered noise characteristics in the generated data (decrease variance only at epoch 11700) (e) Re-train to adapt to new variance (epochs 12000 → 159000).

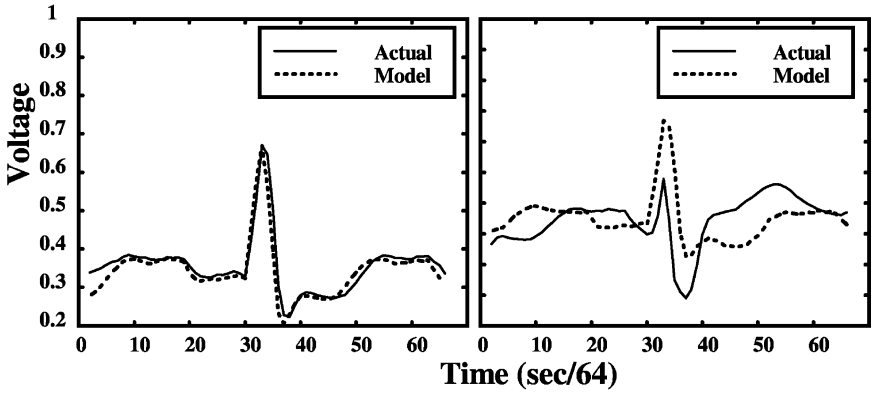

Fig. 7. Heartbeat signal (solid line) for a regular QRS beat (left) and an ectopic beat (right), along with the one-step PoE reconstruction (dashed line) of the same event, from a model with six binary-stochastic experts.

Ectopic beats account for $\approx 1\%$ of the heartbeats in the chosen data. Unsurprisingly, therefore, the PoE models such beats relatively poorly. It is also clear that the PoE models regular beats extremely well. Fig. 7 is encouraging. The PoE captures both the form and the statistics of the data. It is clear from the work described in Tarassenko and Clifford (2001), however, that the MLP network tended to capture the mean of the training set long before capturing the variability associated with normality. It is, therefore, interesting to observe the activity of the PoE beyond the point at which it has apparently, from Fig. 7, 'captured' the training data. The most illuminating method is to ask 'what are the individual experts modelling?', following the approach in Hinton (2000b). Here the features of handwritten digits captured by the individual experts proved to be interesting and 'sensible' receptive fields and sub-features. Fig. 8 shows the features modelled by the individual experts after extended training (after it was judged that no further adaptation would take place).

The average heartbeat is modelled by the biases to the output units. While experts 4–6 model simply noise plus a small addendum to the peak, experts 1, 2 and 3 model fairly major modifications to the peak and side-lobes. Following the results in Tarassenko and Clifford (2001) it is illuminating to

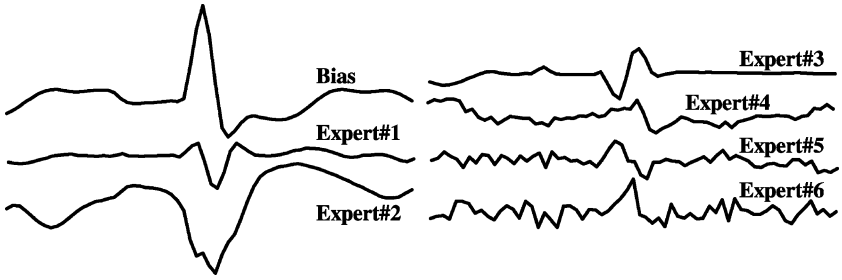

Fig. 8. Data reconstructions from the biases to the output units and six individual experts.

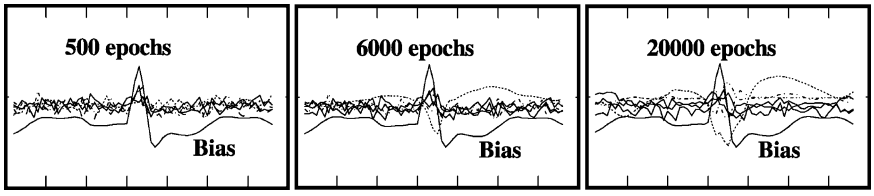

Fig. 9. Evolution of the expert activity during training.

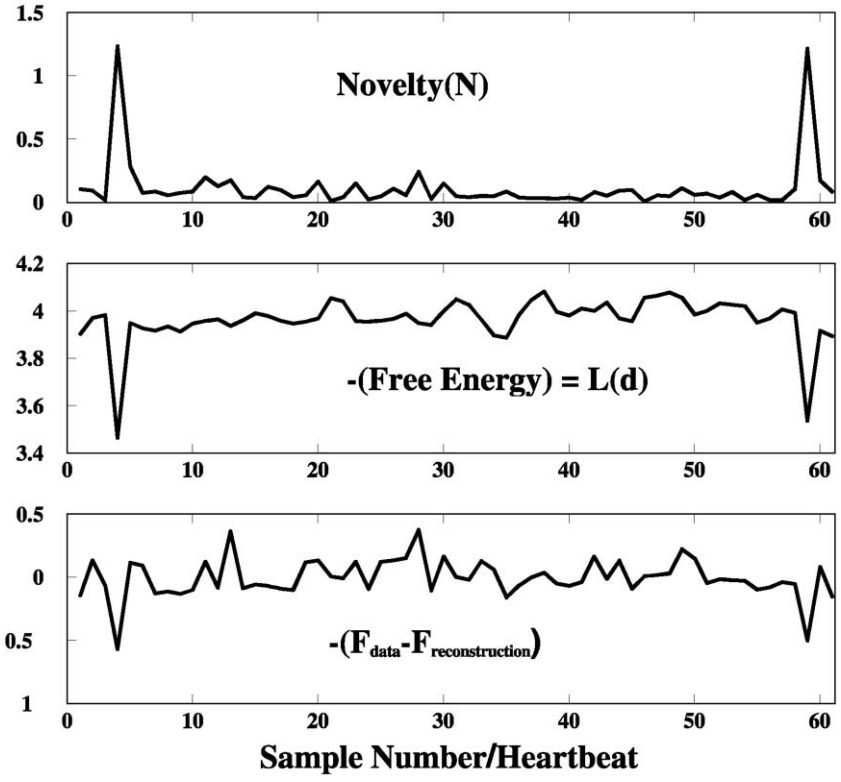

Fig. 10. Novelty signal N, -(Free Energy) or $L(\mathbf{d})$ and the difference between the Free Energies for datum and one-step reconstruction for a 6-expert PoE. The data contains 2 ectopic beats.

comment upon the development of these features as training progresses (Fig. 9). It is difficult to track individual experts in this black-and-white diagram, but it is clear that the 'average heartbeat' is modelled by the bias terms extremely rapidly, while the experts per se merely model noise. At 500 epochs, the model is good as viewed in Fig. 7—but the experts are rather inexpert! As training is continued well beyond the point at which the 'error signal' (in this case, effectively the novelty signal N) has stabilised, the experts become an active part of the model (between epochs 5000 and 20,000), as shown in Fig. 8. It is not yet clear why it takes so long for the experts to become useful. If it is simply attributable to the stochastic nature of the training process, an expert would occasionally become 'active' early in training. This has not been demonstrated. Experiments to improve understanding of this extended-training phenomenon are ongoing. Finally, we will

show that a PoE trained on heartbeats has the potential to be used as a novelty detector.

3.4. Novelty detection: real data

Fig. 10 shows the novelty signal N and free energy for a 6-expert fully-trained PoE, presented with data that contain two ectopic (abnormal and thus novel) beats.

In Tarassenko and Clifford (2001) ectopic beats were excluded from the training database for the auto-associator. The ectopic beats are clear from the novelty signal N, the log-likelihood and the difference between the Free Energies of the datum and its reconstruction (this is the function that training aims to minimise). N is therefore a usable novelty-detector, with a threshold for novelty set according to either results from a validation set or a running-average of the novelty signal in on-line usage. The results in Fig. 10 are taken from a PoE trained on a full database, including movement artifacts and VEBs. Removing VEBs and artifacts during training improves the results from the PoE. It is more useful, however, in the context of an embedded PoE as proposed above, to examine a system where the PoE trains on all of the data in real time. If an oversupply of experts is provided and long training times are allowed, the PoE simply builds a model of ectopic beats into one or more of the experts. Novelty can then be detected via the activations of these experts, but not in the novelty signal N defined above. In this circumstance, a second PoE layer may extract useful statistical information in the latest variables.

4. Conclusions

As claimed in Hinton (1999, 2000b) products of simple experts can build good models of non-trivial data. We have shown how such models can also be useful. Furthermore, PoEs are more suitable for implementation in compact, noisy (potentially deep-sub-micron) analogue hardware than any other 'neural' architecture in our experience. The PoE has been shown to be capable of classification (Hinton, 2000b) and we have shown it to be usable as a novelty detector, with potential applications as an adaptive 'front-end' in integrated sensor systems. The 'behaviour' of the expert variables has been studied in detail and it is clear that the training process causes significant changes in the parameters of the latent variables long after training has apparently 'settled'. We are currently investigating the limitations of the binary-unit PoE in modelling real-valued data, as well as the RBMRate extension reported in Teh and Hinton (2001), developed for that specific purpose. In summary, the neural architectures community has at last given the neural hardware community something that is both implementable and worth implementing.

Acknowledgements

The author is extremely grateful to Geoffrey Hinton for his continued inventiveness, to Yee Whye Teh for direct and vital assistance in getting a PoE working in MatLab, to Lionel Tarassenko and Gari Clifford for heartbeat data and advice and to Dean McNeill for helping me to believe that PoEs work the way they do.

References

Astaras, A., Dalzell, R. W., Murray, A. F., & Reekie, H. (1999). Pulse-based circuits and methods for probabilistic computation. *Proceedings International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems*, 96–102.

Dalzell, P., & Murray, A. F. (1999). A framework for a discrete valued Helmholtz machine. *IEE International Conference on Artificial Neural Networks*, 49–54.

Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz machine. *Neural Computation*, *7*, 1022–1037.

Grossberg, S., & Carpenter, G. (1991). *Pattern recognition by self-organizing neural networks*, Mass.: MIT Press.

Hebb, D. (1949). *The organisation of behavior*, New York: Wiley.

Hinton, G. (1999). Products of Experts. International Conference on Artificial Neural Networks (ICANN), Edinburgh, pp. 1–18.

Hinton, G. (2000). Simpler training procedures for products of experts. Private communication.

Hinton, G. (2000). Training Products of Experts by minimising contrastive divergence. Technical Report: Gatsby Computational Neuroscience Unit (TR2000-004).

Mclachlan, G., & Basford, K. (1988). *Mixture models: inference and applications*, New York: Dekker.

Murray, A. F., & Edwards, P. (1993). Synaptic weight noise during MLP training: fault tolerance and training improvements. *IEEE Trans. Neural Networks*, *4* (4), 722–725.

Murray, A. F., & Tarassenko, L. (1994). *Neural computing: an analogue VLSI approach*, London: Chapman Hall.

Murray, A. F., & Woodburn, R. (1998). The prospects for analogue neural VLSI. *International Journal of Neural Systems*, 559–580.

Smolensky, P. (1986). Information processing in dynamical systems: foundations of harmony theory. In J. L. McLelland & D. Rumelhart, *Parallel distributed processing: explorations in the microstructure of cognition* (pp. 194–281). Cambridge, MA: MIT Press.

Tarassenko, L. & Clifford, G. (2001). Detection of ectopic beats in the electrocardiogram using an auto-associative neural network. Kluwer (to be published).

Taylor, J. & Bressloff, P. (1990). Discrete time models of noisy networks. Proc. International Conference on Neural Nets, Paris.

Teh, Y.-W. & Hinton, G. (2001). Rate-coded restricted Boltzmann machines for face recognition. Advances in Neural Information Processing Systems 13, 908–914.