ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 63-68

Matching analogue hardware with applications using the Products of Experts algorithm

Patrice Fleury, Robin Woodburn and Alan F. Murray

Neural Networks Research Group,
Department of Electrical Engineering,
The University of Edinburgh, King's Buildings,
Mayfield Rd, Edinburgh - EH9 3JL,
Scotland - UK
pcdf,rjw,afm@ee.ed.ac.uk

Abstract. Probabilistic algorithms offer a means of computing that works with the grain of analogue hardware, rather than against it. This paper proposes the use of such an algorithm in applications where the advantages of analogue hardware are most likely to be realised.

1. Introduction

This paper argues for the exploration of new directions in the development of artificial neural networks (ANNs) in hardware, and presents work on one possible direction involving analogue hardware.

Software simulations of ANNs are easy to implement, flexible, accurate, need few compromises in design (provided results are not required in real time) and offer solutions to many applications.
Hardware versions of these networks, on the other hand, offer increased power (parallelism and hence speed) in a reduced space, but at a price. The price is that they are difficult to implement, inflexible, inaccurate, and require many compromises in their realisation [1] [2].

For digital hardware, there is no clear application. Although commercially-available accelerator boards exist, it is arguable that DSPs offer the same order of performance and flexibility at much less cost, while conventional processors "catch up" extremely rapidly.
A well-defined application for analogue hardware is even harder to establish. Analogue designs have some potential benefits, including low power consumption and increased parallelism. Set against this are some clear disadvantages. Popular ANN algorithms, developed in software on digital machines, are difficult to map onto analogue hardware. Extracting the benefits from true parallelism is not easy, and DSPs can often match, or better, analogue performance.

New directions must be explored in our attempts to draw the different threads of algorithms, hardware and applications together. However, we can set down some criteria for what is required. Rather than force existing algorithms to fit the hardware, we must develop algorithms that exploit the natural characteristics of hardware. At least for analogue hardware, niche applications are more likely, involving the processing of raw signals without any intermediate data conversion. Such an application should make the most of the advantages of analogue

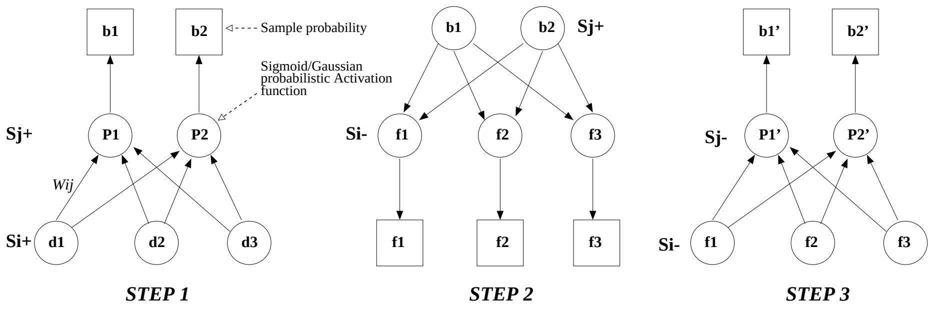

Figure 1: *shows the procedure executed by the learning process. The data* {d} *presented to the input layer defines the probability* {P} *of each Expert using Eq.1. This probability is then sampled* {b} *(STEP 1) and fed down the network so that a fantasy data vector* {f} *is obtained (STEP 2). This latest data vector is then fed up the network again in order to derive to new value* {P'} *of the experts (STEP 3). Ideally the results obtained with the original data set* {d} *and its fantasy vector should be identical. When they are not, the weights* W_{ij} *are changed according to Eq. 2*

circuitry, namely low power consumption and a high degree of parallelism.

The work presented in this paper meets the criteria for bringing algorithms, hardware and applications together. The networks in which we are interested, and their training algorithms, are probabilistic: that is, the inputs to the network do not set outputs deterministically, but control the probabilities of particular outputs. Probabilistic models are a useful representation of processes that involve uncertainty, and that is exactly true of analogue hardware. The application we have in mind – i.e. the recognition of unusual patterns in heartbeat data – is precisely the kind of applications where small size, low power consumption and parallelism could be best exploited.

An additional reason for our interest in this work is that we expect it to influence concurrent projects looking at the probabilistic behaviour of transistors, as processes shrink to deep-sub-micron dimensions.

2. Probabilistic computing

The issue of computation in the presence of uncertainty has attracted considerable interest in the last decade. It certainly is of interest for those whose interests are in implementing ANNs in aVLSI (analogue Very Large Scale Integration). Uncertainties and non-idealities inherent to VLSI processes affect the behaviour of ANNs on silicon and these effects will increase as VLSI device dimensions drop below 100nm. Probabilistic computing offers a new approach for dealing with uncertainty and extracting useful, relevant information or features from noisy environments or corrupted data sets. It also offers a possible way forward for performing useful and accurate computation with noisy and ill-matched components.

2.1. Stochastic Networks

Pioneering work by Pearl [3] brought Bayes' mathematical work to bear upon probabilistic ANNs. This resulted in Bayesian Belief Networks, which proved powerful, but computationally very expensive.
Bayesian inference was then simplified by introducing stochastic sampling methods such as Markov Chain Monte Carlo sampling [4] and the Boltzmann Machine [5]. However, the algorithmic complexity of the above rendered these architectures poor candidates for hardware implementation.
Recently, Hinton introduced the Helmholtz Machine [6], a form of probabilistic network which proved amenable to VLSI implementation [7]. Since then, Hinton has developed the Products of Experts (PoE) algorithm [8], which shares some similarities with the Helmholtz Machine, but has a more reliable and principled training algorithm. As a result, PoEs show even more amenability to hardware implementation.

2.2. PoE : Overview

The PoE is an unsupervised, stochastic ANN comprising a set of probabilistic generative models ("Experts"). The network aims to represent data by a conjunctive mixture of different individual models. In other words, the probability density functions (PDF) of the experts (Gaussian, Sigmoidal or other) are multiplied together, rather than summed [9]. They are then renormalised [8], so that a global model with a more incisive PDF can be obtained.
The conjunctive nature of the network therefore allows a sharper distribution to be obtained. Another equally important property that results from these products is that regions of the data space can be completely, or partially, discarded by some models. In other words, it is acceptable for experts to assign high probability to an invalid region, as long as at least one other Expert vetoes it by assigning it a low probability.
This, in turn, is important as it permits simpler and cruder Expert models to be used, rendering hardware implementation easier.

3. PoE : Some details

3.1. The PoE algorithm

A full explanation of the PoE is given in [10]. Only the final results are presented here. If the experts are binary stochastic units, then the probability of each neuron being "on" is described by :-

$$P(S_j = 1) = \sigma(\sum W_{ij}.S_i) \tag{1}$$

where σ is the activation function (be it Gaussian, Sigmoidal, etc...), W_{ij} the synaptic weights of a neuron, S_i and S_j the states of the input (visible) and hidden layers.
The learning is then defined by the changes made to the synaptic weights in order to minimise the "contrastive divergence" - a measure of model quality based upon one-step reconstructions of the data by the experts. This weight update is calculated using equation 2.

$$\Delta W_{ij} = \epsilon(< S_i^+ S_j^+ > - < S_i^- S_j^- >) \tag{2}$$

where ϵ is the learning rate or step, $< S_i^+ S_j^+ >$ represents the expectation value of the products of S_i and S_j that results when a data vector $\{d\}$ is applied to the input. Similarly, $< S_i^- S_j^- >$) represents the expectation value that results from a fantasy (one-step reconstruction) vector $\{f\}$ being input.

3.2. Algorithmic simplification

The weight update process (also described in Fig.1) is admirably simple and amenable to hardware. A small modification could, however, make its aVLSI implementation even more attractive, with little loss of performance.
Instead of trying to apply synaptic weight change directly, it could be incremented/decremented by fixed-size steps in the direction of the expectation value.

$$\Delta W_{ij} = \epsilon \; sign(< S_i^+ S_j^+ > - < S_i^- S_j^- >) \tag{3}$$

Therefore a weight only changes by a value of $\pm\epsilon$ or 0. It can be seen from Fig.2 that both solutions converge to a result of equal "quality" (not necessarily the same results, as the training is stochastic), although the simplified version does not converge as rapidly.

4. Hardware implementation

Analogue designs are not discussed in detail here (this will be the subject of a future paper). A simple description of the basic elements needed to implement the PoE is, however, presented.
Most of the elements required to perform on-chip learning are straightforward. As eq.3 shows, only multipliers and a subtracter are needed to determine the weight update. However, the actual operation of weight-modification is more difficult as a small change must be made to a weight voltage without introducing extraneous noise or spurious increments/decrements [1]. Our design approach is to isolate the input weight of the weight-modification circuit so that noise and coupling do not alter the input weight. To do so the synaptic weight is "copied" so that a reference weight voltage is obtained. From this reference voltage, two voltages are derived, one slightly larger and one slightly smaller. Finally, only one of these two signals is selected and fed back to the input, according to the learning rule.
The other major element is to introduce stochasticity into the network. Therefore a binary stochastic neuron (Fig.3) obeying equation 1 is required. The stochasticity is realised by an asynchronous voltage-controlled oscillator. The net ("squashed") input to a neuron determines the probability that it is "on". This probability is represented by a voltage which controls the pulse width:period ratio of an oscillator. Therefore the probability of neuron activation is encoded in the duty cycle of the oscillations. Now, when the output waveform of the oscillator is sampled, the correct form of probabilistic behaviour results, provided that the oscillation frequency and sampling frequency are not correlated.

5. Some applications

5.1. Ectopic heartbeat

Cardiovascular diseases, inducing sudden cardiac death (SCD), are a common cause of mortality. Abnormalities in heart rate signals (in the form of ectopic heartbeats) are believed to be predictive of SCD. Rapid, on-line detection of such beats would therefore be a useful diagnostic tool.

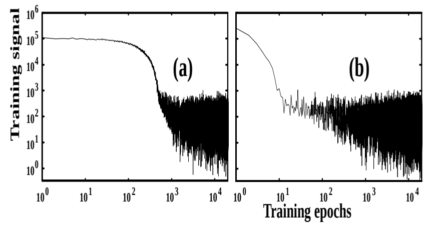

Figure 2: *This shows different training of the experts, (b) shows a training using eq.2, while (a) is using a simplified version, eq.3.*

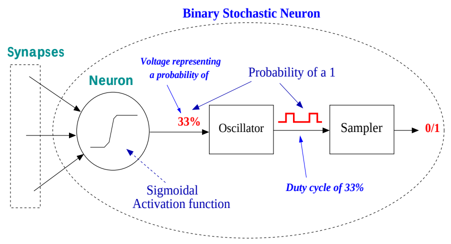

Figure 3: *Shows how the stochasticity of the network is obtained.*

The advantages of a compact, low power device, directly interfaceable to the body, that would detect cardiac abnormalities are clear.
Real data, containing 1% of ectopic beats (EB) was used to train a PoE of 6 experts [11]. Therefore, unsurprisingly enough, the PoE models the EB beats less well than it does the regular beats (Fig.4). Removing EB and artifacts from the training data set "improves" the results, as the PoE would have no opportunity to model abnormal signals. In contrast, a large number of experts and long training times results in the EB being included in the PoE model. In this scenario, a different approach to novelty detection must be taken, as the PoE has, effectively, become a two-class classifier, trained with one class (EB) very under-represented.

5.2. Novelty detection

Once a PoE network is trained and the training disabled, only the type of data on which the PoE was trained is regenerated well. Data outside the expected regions is not well modelled. This can be observed in Fig.4 where the PoE does not produced a good reconstruction of the EB. During training, the reconstruction error drives weight adjustment. The amount by which the weights change (Eq.2) is therefore representative of the mismatch between the current datum and the model and thus the degree of "novelty". Therefore if we were to disable the training by setting the learning rate (ϵ) to zero, a good indication of "novelty" would be given by the contrastive divergence i.e. by eq.2 if ϵ was ignored. An overall novelty signal (Eq.4) can then be obtained by calculating the average synaptic weight change across the entire PoE.

$$N = \sum_{ij} (< s_i^+ s_j^+ > - < s_i^- s_j^- >) \tag{4}$$

A set of data containing two EB was presented to the previously-trained PoE of 6 experts. The resulting novelty signal (Fig.5) does, indeed, peak when abnormal data are present.

6. Conclusion

Probabilistic neural networks are interesting from the perspective of aVLSI for a number of reasons. A particular stochastic architecture, the Product of Experts, has been shown to be capable of building good models of some real and artifical data, as well as being amenable to hardware implementation. We have also

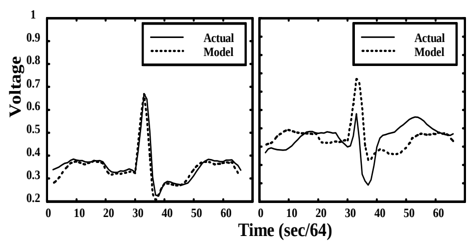

Figure 4: *(Left) Signal of a regular heart beat along with an ectopic heartbeat (Right). The dashed lines represent the one-step PoE reconstitution of the same heartbeats. The large positive value at time 0 represents a bias value.*

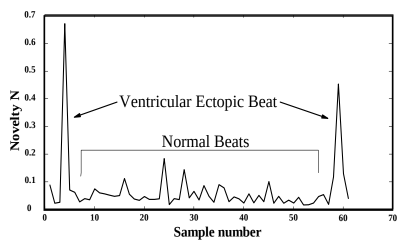

Figure 5: *Novelty signal for a 6-Expert PoE, presented with data containing two ectopic beats.*

indicated why an analogue hardware implementation of the PoE is not purely of academic interest, but has also great potential for specific real world monitoring applications.

References

[1] Lehmann T., Woodburn R., and Murray A.F., "On-chip learning in pulsed silicon neural networks," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, 1997.

[2] Woodburn R. and Murray A.F., "Implementing artificial neural networks in analogue VLSI," in *Proceedings of the 4th International Conference on Neural Information Processing*, Dunedin, New Zealand, 1997, pp. 658–661.

[3] J.Pearl, "Fusion, propagation and structuring in Belief Networks," *Artificial intelligence*, vol. 19, no. 3, pp. 241–288, 1986.

[4] R.M. Neal, "Connectionnist learning of Belief Networks," *Artificial intelligence*, vol. 56, pp. 71–113, 1992.

[5] G.E. Hinton and T.J. Sejnowski, "Parallel distributed processing: Explorations in the microstructure of cognition," *MIT Press, Cambridge, MA, USA*, vol. 1, pp. Learning and relearning in Boltzmann Machines, 1986.

[6] Dayan P., Hinton G. E., Neal R., and Zemel R. S., "The Helmholtz Machine," *Neural Computation*, vol. 7, no. 5, pp. 889–904, Sept. 1995.

[7] A. Astaras, R. Dalzell, A.F. Murray, and M. Reekie, "Pulse-based circuits and methods for probabilistic neural computation," in *Microneuro '99 IEEE Conference*, April 1999, Granada, Spain.

[8] G.E. Hinton, "Products of Experts," in *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN'99)*, Edinburgh, Scotland, Sept. 1999, pp. 1–6.

[9] Ueda N., Nakano R., Ghahramani Z, and Hinton G.E., "SMEM algorithm for mixture models," *Neural Computation*, 1999.

[10] Hinton G.E, "Training Products of Experts by maximizing contrastive likelihood," Tech. Rep., Gatsby Computational Neuroscience Unit, 1999.

[11] A.F. Murray, "Products of (simple) Experts: Hardware possibilities and new applications," *Dept. of Electronics and Electrical Eng., University of Edinburgh*, April 2000, http://www.ee.ed.ac.uk/˜afm.

[12] R.Woodburn, A. Astaras, R. Dalzell, A.F. Murray, and D. McNeill, "Computing with uncertainty in probabilistic neural networks on silicon," in *the 2nd International ICSC Symposium on Neural Computation (NC'2000)*, Berlin, Germany, May 2000.