Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006

Training Probabilistic VLSI models On-chip to Recognise Biomedical Signals under Hardware Nonidealities

P.C. Jiang, and H. Chen, *Member, IEEE*

***Abstract*—VLSI implementation of probabilistic models is attractive for many biomedical applications. However, hardware non-idealities can prevent probabilistic VLSI models from modelling data optimally through on-chip learning. This paper investigates the maximum computational errors that a probabilistic VLSI model can tolerate when modelling real biomedical data. VLSI circuits capable of achieving the required precision are also proposed.**

I. INTRODUCTION

VLSI implementation of probabilistic models has been attractive for many biomedical applications, owing to the fact that probabilistic models are able to use stochasticity to generalize natural variations in real data, and that VLSI implementation facilitates the miniaturization of implantable devices in biomedical application. Among the proposed probabilistic VLSI models[1-3], the VLSI system based on the Continuous Restricted Boltzmann Machine(CRBM) [4] is of particular interest, because the CRBM system is consisted of "probabilistic" component circuits whose inputs merely decide output probability. This characteristic is potentially useful for enhancing the robustness against noise or computational errors in analogue circuits. To exploit this characteristic, on-chip adaptability is essential for the CRBM system to find optimum levels of stochasticity so as to generalise noise and errors. However, experiments in [4] indicated that the CRBM system still requires precise training circuits to achieve optimum modelling. By representing the effect of hardware nonidealities as an offset in the training algorithm, this work investigates the maximum training offset that the CRBM system can tolerate to model real biomedical(ECG) data. In addition, circuits capable of achieving the required precision are proposed.

II. THE CRBM SYSTEM

The CRBM system refers to the VLSI implementation of the Continuous Restricted Boltzmann Machine, a probabilistic model containing two layers of continuous-valued, stochastic neurons with interlayer connections[5], as shown in Fig.1. Circle v_i represents visible neurons each of which corresponds to one dimension of data the CRBM models.

P.C. Jiang and H. Chen are with the Institute of Electronics Engineering, the National Tsing-Hua University, HsinChu, 30013 Taiwan (Tel/Fax: 886-3-5162221; e-mail: g935011@oz.nthu.edu.tw, hchen@ee.nthu.edu.tw).

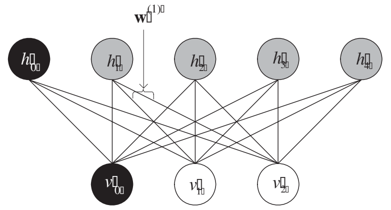

Fig.1 : The architecture of a CRBM with two visible and four hidden neurons. v_0 and h_0 represent bias units with invariant outputs v_0=h_0=1.

Circle h_j represents hidden neurons which function as individual experts that extract features underlying the data. Let s_i denote the state of neuron v_i or h_j, and w_{ij} the bidirectional connection weight between v_i and h_j. The stochastic behaviour of neurons is described by [5]

$$s_i = \varphi\left(a_i \cdot \left(\sum\nolimits_j w_{ij} s_j + N(\sigma,0)\right)\right) \tag{1}$$

where $N(\sigma,0)$ represents a zero-mean Gaussian noise with variance σ, and $\varphi(\cdot)$ a sigmoid function with asymptotes at ±1 and slope controlled by a_i.

As a *generative* model, the CRBM aims to "regenerate" training data distributions in its visible neurons. Testing data can then be categorized according to the responses of hidden neurons of a trained CRBM [5]. In the CRBM system, parameters are trained according to the following simplified rule [4]

$$\Delta\lambda = \eta_\lambda \cdot \left(< v_i \cdot h_j >_4 - < \hat{v}_i \cdot \hat{h}_j >_4 \right) \tag{2}$$

where λ represents both w_{ij} and a_i, η_λ the updating rate, and ($<v_ih_j>_4$ - $<v_ih_j>_4$) the contrastive divergence between initial samples and one-step samples of v_i and h_j [4]. $<\cdot>_4$ denotes taking the expectation over four training data. Approximate equilibrium distribution of a trained model can be obtained by Gibbs sampling visible units for multiple steps[5]. The similarity between the multi-step reconstruction and the training data tells how well a CRBM models a dataset.

III. MODELLING ARTIFICIAL DATA UNDER HARDWARE NONIDEALITIES

The precision of analogue circuits is limited by process variations, charge injection, and device nonlinearity, among which process variations are normally the dominant factor. Experiments in [4] has shown that the overall effect of hardware nonidealities in training circuits can be modeled as

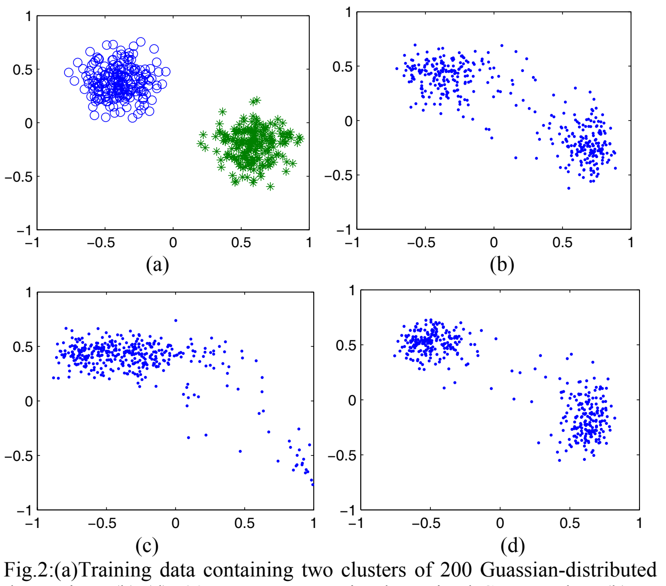
Fig.2:(a)Training data containing two clusters of 200 Guassian-distributed data points. (b)-(d): 20-step reconstruction by trained CRBM when (b) Δ_T =0 (c) Δ_T =2% (d) Δ_T =1%

the following "biased" training algorithm

$$\Delta\lambda = \eta_\lambda \cdot sign\left(\left\langle v_i h_j \right\rangle_0 - \left\langle v_i h_j \right\rangle_1 + \Delta_T\right) \tag{3}$$

where Δ_T represents the offset introduced by hardware nonidealities. The existence of Δ_T limits the minimum divergence achievable by on-chip training circuit, preventing the CRBM system from modelling data optimally.

In this paper, the offset value is given in terms of percentage, normalized with respect to the maximum absolute value of $<v_i h_j>_4$ (i.e.Δ_T=1% refers to Δ_T=0.01 in (3)). Based on the mapping between hardware and Matlab simulation derived in [4], the behaviour of a large-scale CRBM system in the presence of hardware nonidealities was simulated in Matlab.

A CRBM system with two visible and four hidden units was first trained to model a two-cluster Gaussian-distributed data, as shown in Fig.2(a), with training rule in (3) and η=0.005 for 40,000 training epochs. Without offset (Δ_T=0), the trained CRBM system reconstructed data distribution as shown in Fig. 2(b). The agreement between Fig.1(a) and (b) indicates that an optimum modelling was achieved. With Δ_T>1%, however, the CRBM system reconstructed data as shown in Fig.2(c). A subset of reconstructed data condenses at the bottom-right corner, indicating that the presence of offset caused several w_{ij} [4] to have a large biased value and thus to discourage optimal modelling [4]. With Δ_T=1%, the CRBM system reconstructed the data distribution shown in Fig. 2(d), indicating that the tolerable offset equals 1%. This result agrees with the simplest experiment in [4], implying that the CRBM can tolerate a maximum of 1% errors when modelling simple but nontrivial two-dimensional data. The precision required for modelling high-dimensional, complex data is futher examined in the context of recognising real ECG data.

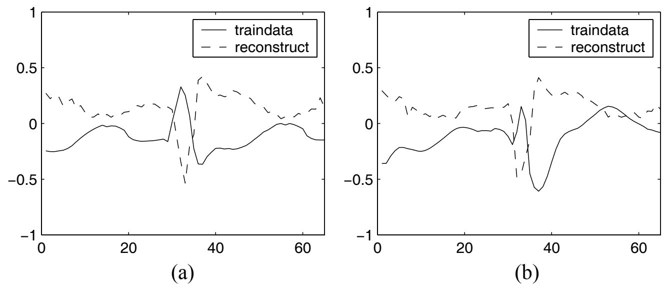
Fig. 3. Results of training the CRBM with Δ_T= 1% (a)The normal heartbeat in the training dataset (solid) and the reconstruction by the trained CRBM(dashed) (b) The abnormal heartbeat in the training dataset (solid) and the reconstruction by the trained CRBM (dashed).

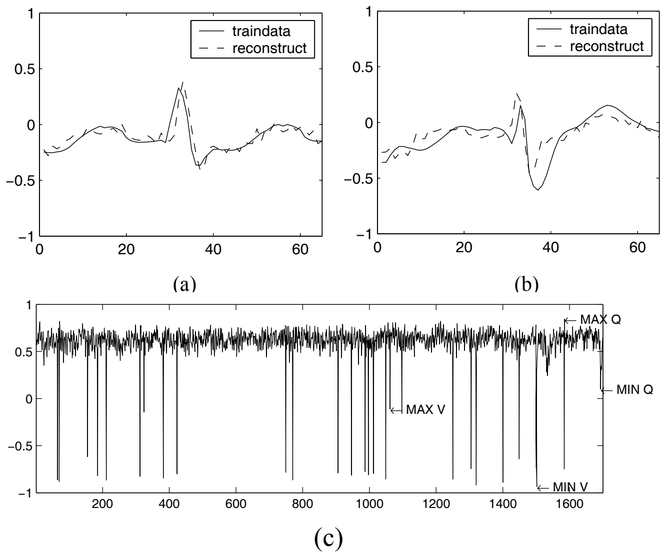

Fig.4: Results of training the CRBM with Δ_T= 0.1% (a)The normal heartbeat in the training dataset (solid) and the reconstruction by the trained CRBM(dashed) (b) The abnormal heartbeat in the training dataset (solid) and the reconstruction by the trained CRBM (dashed). (c) Responses of hidden neuron H3 to 1700 testing data. Any threshold between MaxV and MinQ can detect abnormal heartbeat with 100% accuracy.

IV. Modelling Biomedical Data under Hardware Nonidealities

The ECG data is obtained from the MIT-BIH database and extracted into training and testing datasets as in [5]. The training dataset contains 500 heartbeats with only 5 abnormal heartbeats. The testing dataset contains 1700 heartbeats with 27 abnormal heartbeats. Each heartbeat is sampled as a 65-dimensional datum. A CRBM system with 65 visible and four hidden neurons was trained to model the training dataset under hardware nonidealities, i.e. with (3) and η=0.005 for 8,000 epochs.

Without offset (Δ_T=0), [5] has shown that the CRBM is capable of detecting abnormal heartbeats in testing data with 100% accuracy reliably. The maximum offset a CRBM system can tolerate is thus identified based on whether abnormal heartbeats can be detected with 100% accuracy. With Δ_T =1%, the trained CRBM system regenerated normal and abnormal heartbeats as shown in Fig.3(a) and (b), respectively. The weights of the trained model, though not shown, were found to be severely "biased" with an offset of

Table.1 : The maximum offsets that a CRBM system can tolerate to model ECG data with different training algorithms.

Training method	Tolerable offset
Four-data, sign-valued update	0.1%
Four-data, real-valued update	0.3%
Single-datum, real-valued update	0.1%

about 30, resulting in the biased reconstruction in Fig.3. Further simulation shows that the CRBM system can tolerate an offset of only 0.1%. Fig.4(a) and (b) show the reconstruction generated by the trained model with Δ_T =0.1%. The responses of hidden neuron h3 to 1700 testing data is shown in Fig.4(c), showing that the trained CRBM system is able to detect abnormal heartbeats with 100% accuracy.

As implementing training circuits with an offset less than 0.1% is quite a challenge, it is important to explore training strategies capable of releasing this strict requirement. The training strategies considered include (a)updating parameters with real-valued contrastive divergence ($<v_ih_j>_0$ - $<v_ih_j>_1$) instead of taking only its sign (b)updating parameters with real-valued contrastive divergence datum by datum without accumulating opinions from four data. The modified training algorithms for (a) and (b) are described by (4) and (5), respectively, as follows.

$$\Delta\lambda = \eta_\lambda \cdot \left(< v_i \cdot h_j >_4 - < \hat{v}_i \cdot \hat{h}_j >_4 + \Delta_T \right) \tag{4}$$

$$\Delta\lambda = \eta_\lambda \cdot \left(v_i \cdot h_j - \hat{v}_i \cdot \hat{h}_j + \Delta_T \right) \tag{5}$$

The reason for considering real-valued update is that real-valued update could allow the CRBM system to correct mislead updates more efficiently. The work in [6], on the other hand, shows that offsets in training circuits mainly come from accumulators. The possibility to avoid accumulating opinions from four data is thus also tested. Table.1 summarizes the maximum tolerable offsets for training algorithms (3), (4), and (5). The four-data, real-valued update does enhance the tolerance against offsets slightly as expected, though an offset smaller than 0.3% remains a challenge. The maximum tolerable offset of single-datum training, on the other hand, is the same as that of four-data, sign-valed training. This indicates that single-datum update does not degrade modelling capability as long as real-valued update is employed.

V. Circuit Topology And Simulation Results

The work in [6] shows that offsets in training circuits mainly come from accumulators that calculate the expectation $<\cdot>_4$ in (3), and subtractors that calculate the difference between the two expected values in (3). As errors in the two circuits are mainly introduced by threshold-voltage mismatches in current mirrors [6], we propose the circuit in Fig.5 that uses dynamic current mirrors (DCM) [7] to minimize the errors. The proposed circuit contains one NMOS (M1-M1c) and

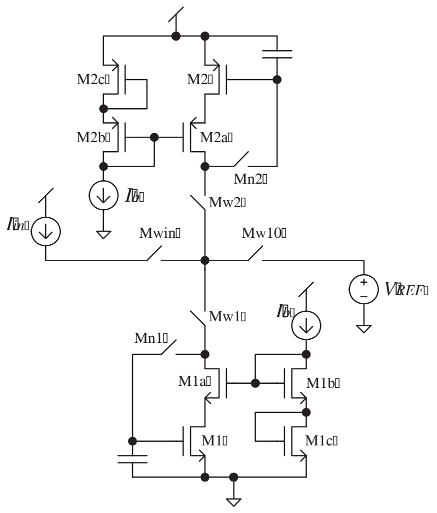

Fig. 5: The circuit that both implements training rule in (5) and function as an accumulator for training rules in (3) and (4)

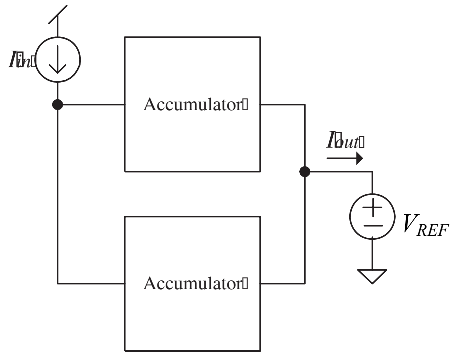

Fig.6 : The circuit architecture that implements the training algorithms in (3) and (4).

one PMOS (M2-M2c) dynamic current mirrors. Each DCM works as a register, using the same transistor (M1 or M2) to copy currents from input to output, and thus to avoid mismatching errors. I_{in} in Fig.5 represents a current output from a multiplier, corresponding to v_ih_j or $\hat{v}_i\hat{h}_j$ in (3). At the first step, switches Mwin, Mw1, and Mn1 are closed to sample the current representing v_ih_j into the NMOS DCM. The current is then stored as a voltage across the capacitor C1 after Mn1 and Mwin become opened. At the second step, switches Mw2 and Mn2 are closed to copy the same current into the PMOS DCM. With Mw2 and Mn2 opened subsequently, the current is stored as a voltage across the capacitor C2. At the third step, let I_{in} correspond to $\hat{v}_i\hat{h}_j$. The current is sampled and stored in the NMOS DCM by repeating the first step. Finally, closing Mw1, Mw2, and Mw10 gives an output current proportional to (v_ih_j- $\hat{v}_i\hat{h}_j$). This is the contrastive divergence in the single-datum training algorithm in (5).

To implement the four-data training algorithm in (3) and (4), the circuit in Fig.5 is used to accumulate opinions of multiple data rather than calculating contrastive divergence. The first two steps described above are carried out once to store v_ih_j of the first datum into the PMOS DCM. At the third step, let I_{in} correspond to v_ih_j of the second datum. Repeating the first step with Mw2 closed stores the *sum* of v_ih_j of both data into the NMOS DCM. Repeating the second and the third steps alternatively then sums up v_ih_j of multiple data and stores it into the NMOS DCM. To calculate the contrastive divergence in (3) and (4), two identical accumulators are employed as in Fig.6 to calculate $\langle v_ih_j\rangle_4$ and $\langle\hat{v}_i\hat{h}_j\rangle_4$, respectively. The top accumulator calculates $\langle v_ih_j\rangle_4$ and stores the value into to its PMOS DCM by running the second step once more, while the other one simply stores $\langle\hat{v}_i\hat{h}_j\rangle_4$ into NMOS DCM. As soon as switch Mw10 is closed, outputs of the two accumulators are connected together to produces an output current proportional to $\langle v_ih_j\rangle_4-\langle\hat{v}_i\hat{h}_j\rangle_4$. This current value can be used to implement either (3) by adding a current comparator at output, or (4) by using the current to charge or discharge capacitors which store parameter values.

The proposed circuits are designed to be fabricated with the TSMC 0.35um 2P4M CMOS process, and the performance of the circuit is simulated in HSPICE. For single-datum update, I_{in} is designed to range from 1μA to 3μA, corresponding to v_ih_j=-1 and v_ih_j =1, respectively. To minimize the dependence of charge injection on I_{in}, the charge-injection error is minimized at I_{in}=2μA. Table.2 shows the simulated results of using the circuit in Fig.5 to calculate the contrastive divergence in (5) at different current levels. The required precision is achieved, but slightly unsatisfactory errors occur at the extreme values of I_{in}. This error comes from the dependence of charge injection on I_{in}, and can be further reduced by slowing down the speed of switching. (t_r=t_f=5ns in our simulation).

The performance of the circuit in Fig.5 in accumulating four data was also simulated. In this experiment, I_{in} is designed to range from 0.25μA to 0.75μA, such that the accumulation of four data still ranges from 1μA to 3μA. This allows the minimization of charge-injection error to remain at I_{in}=2μA. Table.3 shows the simulation results of accumulating currents with various levels. The errors are all greater than the precision(0.3%) required for four-data training, demonstrating again that errors mainly come from accumulation. Nevertheless, the symmetrical architecture of the circuit in Fig.6 allows the accumulating errors to cancel each other. The performance of the circuit in Fig.6 in calculating the contrastive divergence of (3) and (4) was summarized in Table.4. Subtracting $\langle v_ih_j\rangle_4$ by $\langle\hat{v}_i\hat{h}_j\rangle_4$ cancels part of charge-injection errors, such that required precision is achieved in the first three experiments. Complete cancellation is infeasible because one accumulator requires one more step than the other to transfer $\langle v_ih_j\rangle_4$ into the PMOS DCM. If the outputs of these two accumulators are sent in parallel into a current comparator, an even smaller error can be achieved. The fourth experiment, however, displays an unsatisfactory result, indicating that charge-injection errors are dependent on the sequence of currents to be accumulated.

Table.2: Single-datum, real-valued update calculated by the circuit in Fig.5

v_ih_j	$\hat{v}_i\hat{h}_j$	Ideal	Simulation	Error
1μA	1μA	0μA	0.0015μA	0.15%
2μA	2μA	0μA	0.00083μA	0.08%
3μA	3μA	0μA	0.0035μA	0.12%

Table3 Four-data accumulation calculated by the circuit in Fig.5

$\langle v_ih_j\rangle_4$ (μA)	Ideal	Simulation	Error
0.25+0.25+0.25+0.25	1μA	0.99899μA	0.404%
0.5+0.5+0.5+0.5	2μA	2.00216μA	0.864%
0.75+0.75+0.75+0.75	3μA	3.00268μA	1.072%

Table.4: Four-data, real-valued update calculated by the circuit in Fig.6

$\langle v_ih_j\rangle_4$ (μA)	$\langle\hat{v}_i\hat{h}_j\rangle_4$ (μA)	Ideal	Error
0.25+0.25+0.25+0.25	0.25+0.25+0.25+0.25	0(μA)	0.16%
0.5+0.5+0.5+ 0.5	0.5+0.5+0.5+0.5	0(μA)	0.25%
0.75+0.75+0.75+0.75	0.75+0.75+0.75+0.75	0(μA)	0.36%
0.25+0.25+0.75+0.75	0.75+0.75+0.25+0.25	0(μA)	1.8%

VI. Conclusion

Using the Continuous Restricted Boltzmann Machine as an example, we have shown that a probabilistic VLSI model can tolerate training errors less than 0.1% to model real biomedical data optimally. The tolerance can be slightly improved by real-valued, while single-datum training does not degrade the tolerance. Circuits able to implement three training strategies are thus proposed and compared. Simulation results demonstrate that the required precision is achievable by employing dynamic current mirrors. However, calculation of contrastive divergence for four-data training algorithm displays sequence-dependent accumulation errors. Single- datum, real-valued update with slow-switching clocks is therefore a favourable on-chip training method. The proposed circuits has been submitted for fabrication. The measurement results will be presented in the conference.

Acknowledgment

The authors would like to acknowledge the TSMC and the CIC for the fabrication of the chip, and the National Science Council (NSC) in Taiwan for funding this project.

References

[1] Alspector, J., Allen, R. B., Hu, V., and Satyanarayana, S. Performance of a Stochastic Learning Microchip. 1, 748-760. 1989. *Advances in Neural Infornation Processing Systems (NIPS)*

[2] Hsu, D., Bridges, S., Figueroa, M., and Diorio, C. Adaptive Quantization and Density Estimation in Silicon. 15. 2002. *NIPS*

[3] Genov, R. and Cauwenberghs, G. Kerneltron: support vector "machine" in silicon. 14, 1426-1434. 2003. IEEE Trans. on Neural Networks.

[4] Chen H., Fleury P.C.D., and Murray A.F., "Continuous-Valued Probabilistic Behavior in a VLSI Generative Model," *IEEE Trans. on Neural Networks : In Press*, no. 99, pp. 1-16, 2006.

[5] Chen, H. and Murray, A. F., "Continuous restricted Boltzmann machine with an implementable training algorithm," *IEE Proc.-Vision ,Image and Signal Processing*, vol. 150, no. 3, pp. 153-158, 2003.

[6] Fleury, P., Chen, H., and Murray, A. F. On-Chip Contrastive Divergence Learning in analogue VLSI. 3, 1723-1728. 2005. *Budapest*, 2004IEEE,Proceedings of the International Joint Conference on Neural Networks . 7-25-0040.

[7] Wegmann, G. and Vittoz, E. A., "Analysis and improvements of accurate dynamic current mirrors," *Solid-State Circuits, IEEE*, vol. 25, no. 3, pp. 699-706, June1990.