Analog Integrated Circuits and Signal Processing, 30, 149–157, 2002

VLSI Implementation of Fuzzy Adaptive Resonance and Learning Vector Quantization

JEREMY LUBKIN AND GERT CAUWENBERGHS

Electrical and Computer Engineering, Johns Hopkins University, Baltimore MD 21218
E-mail: gert@jhu.edu

Received January 25, 2000; Accepted March 21, 2000

Abstract. We present a mixed-mode VLSI chip performing unsupervised clustering and classification, implementing models of Fuzzy Adaptive Resonance Theory (ART) and Learning Vector Quantization (LVQ), and extending to variants such as Kohonen Self-Organizing Maps (SOM). The parallel processor classifies analog vectorial data into a digital code in a single clock, and implements on-line learning of the analog templates, stored locally and dynamically using the same adaptive circuits for on-chip quantization and refresh. The unit cell performing fuzzy choice and vigilance functions, adaptive resonance learning and long-term analog storage, measures 43 μm $\times$ 43 μm in 1.2 μm CMOS technology. Experimental learning results from a fabricated 8-input, 16-category prototype are included.

Key Words: learning on silicon, vector quantization, adaptive resonance, analog memory

1. Introduction

Adaptive Resonance Theory (ART) [1] is a class of neurally inspired models of how the brain performs clustering and classification of sensory data, and associations between the data and representations of concepts. The models perform unsupervised learning of categories under continuous presentation of inputs, through a process of 'adaptive resonance' in which the learned patterns adapt only to relevant inputs, but remain stable under irrelevant or insignificant inputs.

More or less under the same umbrella, Vector Quantization (VQ) [2] is a commonly used technique for digital encoding of continuous-valued vectorial signals, mostly for applications of data compression and pattern recognition. As with VQ, Fuzzy Adaptive Resonance [3] operates on analog vectorial data, and learns categories stored in the form of analog vector templates. Other variants on unsupervised clustering and vector quantization include Kohonen Self-Organizing Maps (SOM) and Kanerva associative memories, among others.

In its basic form, the implementation of Fuzzy ART and VQ involve a search among a set of vector templates for the one which best matches the input vector, according to a given distance metric or 'choice function.' Learning continually adjusts the best matching templates towards the input, conditional on a 'vigilance' criterion in the case of Fuzzy ART. The process of adaptive resonance for stable preservation of categories under variable inputs during learning is the main difference between ART and other forms of VQ.

Several versions of vector quantizers and their variants have been developed in analog and mixed-mode VLSI, e.g. [12–21]. The 8-input, 16-category chip presented here is implemented in current-mode CMOS technology for low-power operation, and integrates learning as well as long-term dynamic capacitive storage of the analog templates using an incremental partial refresh scheme [9]. The chip is configured to operate either in Fuzzy ART or VQ mode, and can be extended to perform Kohonen SOM through external addressing. The compact size of the unit cell, $71\lambda \times 71\lambda$ in MOSIS scalable CMOS technology, allows to expand the architecture for applications requiring several hundreds of inputs and/or categories integrated on a single chip.

2. Fuzzy Adaptive Resonance Theory

For a detailed exposition of the algorithm, we refer to [3], and also [4], where several variants of Fuzzy ART have been presented. Here we focus on the implemented form, and define the equations with our

notation used to represent the signals. We assume an N-dimensional analog input $\mathbf{I}$, and M weight templates $\mathbf{z}_i$ of same dimension, each representing one category.

2.1. Category Selection

The *choice function* T_i for each category i is computed as

$$T_i = \frac{|\mathbf{I} \wedge \mathbf{z}_i|}{\alpha + |\mathbf{z}_i|} \tag{1}$$

where the fuzzy min operator $\wedge$ is defined componentwise as

$$I_j \wedge z_{ij} = \min(I_j, z_{ij}) \tag{2}$$

Inputs to the classifier are complement-encoded, supplying the complements $\overline{I_j} = I_{\max} - I_j$ along with all inputs I_j. Explicitly,

$$|\mathbf{I} \wedge \mathbf{z}_i| = \sum_i (\min(I_j, z_{ij}) + \min(\overline{I_j}, z'_{ij})) \tag{3}$$

where the template weights for the complementary input components z'_{ij} are independent from z_{ij}. The complement-encoding of the inputs provides automatic normalization, $|\mathbf{I}| \equiv N \cdot I_{\max}$.

Only when a category meets a *vigilance condition,* it enters competition for the highest choice function T_i, which identifies the winning output category. The vigilance condition for each category is formulated as

$$\frac{|\mathbf{I} \wedge \mathbf{z}_i|}{|\mathbf{I}|} > \rho \tag{4}$$

where ρ is the vigilance parameter.

In our implementation, the vigilance condition is checked before a category enters the competition, eliminating the need to iterate the search until a vigilant winning category is found. The only complication arises in the case where there are no vigilant categories. This case triggers the creation of a new category with *fast learning* which replaces one of the existing categories. This is necessary in hardware, as the number of categories is hardwired.

2.2. Learning

Once the winning category has been selected, the weights belonging to that category are updated according to the learning rule:

$$\Delta \mathbf{z}_i = \beta(\mathbf{I} \wedge \mathbf{z}_i - \mathbf{z}_i) \tag{5}$$

If no category is vigilant to enter the competition, a new category is created (or existing category replaced), and initialized with *fast learning,* $\mathbf{z}_i \equiv \mathbf{I}$.

Expanding the fuzzy min operator, the weight updates (5) can be expressed as

$$\Delta z_{ij} = \lambda(I_j, z_{ij})(I_j - z_{ij}) \tag{6}$$

where

$$\lambda(I_j, z_{ij}) = \begin{cases} 1 & \text{for initial coding and recoding} \\ \beta & \text{when } I_j \le z_{ij} \\ 0 & \text{when } I_j > z_{ij} \end{cases} \tag{7}$$

This modulation of the learning rate λ is easier to implement in hardware, since it reduces to directly modulating the current supplying the weight updates.

Note that after coding/recoding the update can only decrease the stored weight value, so it is important that all weights be initialized to their maximum value.

3. Fuzzy ART and VQ Architecture

The chip presented here, a 8×16 Fuzzy–ART classifier and vector quantizer (VQ), is implemented in current-mode CMOS technology for low-power operation, and integrates learning as well as long-term dynamic capacitive storage of the analog templates using an incremental partial refresh scheme [9]. A general overview of VLSI methods used for on-chip learning, and examples of other systems, are presented in [5].

The architecture of the hybrid implementation is shown in Fig. 1. The core contains an array of 8×16

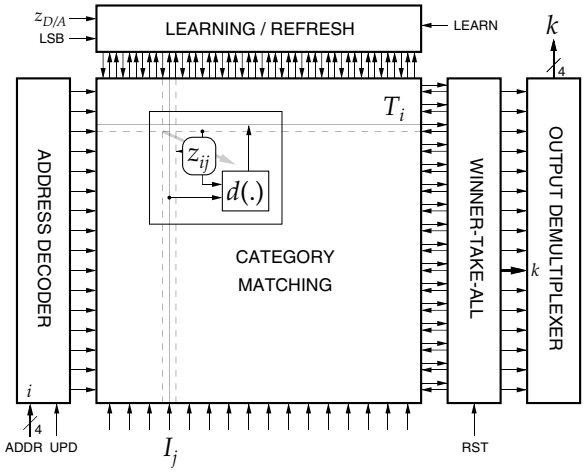

Fig. 1. Parallel VLSI architecture for Fuzzy ART and VQ, including template learning and refresh functions.

template matching cells interconnecting rows of templates with columns of input components. Each cell constructs a distance $d(I_j, z_{ij})$ between one component I_j of the input vector $\mathbf{I}$ and the corresponding component z_{ij} of one of the template vectors $\mathbf{z}_i$. For Fuzzy ART, this is the Fuzzy min as defined above; for VQ the distance is the absolute difference $|I_j - z_{ij}|$, [7]. The component-wise distance is accumulated across inputs along template rows to construct T_i, and presented to a winner-take-all (WTA), which selects the single winner

$$k = \arg \max_i T_i \tag{8}$$

In the case of VQ, T_i is constructed by accumulating $d(I_j, z_{ij})$ without weight normalization and vigilance conditioning.

3.1. Choice Function

A simplified version of the Fuzzy ART choice function is used here, to reduce the hardware implementation. The simplifications are similar to the VLSI-friendly version of the ART1 choice function in [12], which eliminates the need to divide analog signals for computing the choice functions. Details are given in the Appendix.

The approximation amounts to expressing the choice function as a linear combination of fuzzy min and fuzzy max distances between input and template row. In particular, the chip computes in parallel:

$$\begin{aligned} T_i^+ &= |\mathbf{I} \vee \mathbf{z}_i| \\ T_i^- &= |\mathbf{I} \wedge \mathbf{z}_i| \end{aligned} \tag{9}$$

where T_i^- is used for the vigilance condition, and T_i^+ (or $(1-\alpha')T_i^+ - \alpha' T_i^-$) is used for the choice function, which enters competition for the minimum value when the vigilance condition is met. The approximation is nearly perfect close to resonance, and errors become significant only far from resonance, where they are typically masked by a failing vigilance condition.

When the chip is configured in VQ mode, the signals T_i^+ and T_i^- are combined differentially to construct the mean absolute difference (MAD) distance [17]

$$T_i^+ - T_i^- = |\mathbf{I} - \mathbf{z}_i| \tag{10}$$

3.2. Learning and Refresh

Learning is performed by selecting the winning template k and producing an incremental update $\Delta \mathbf{z}_k$ in the stored vector $\mathbf{z}_k$ towards the input vector, according to a modified version of (6). The learning rate λ is modulated according to (7), except in VQ operation for which the learning rate is constant, $\lambda \equiv \beta$. In the case of Kohonen self-organizing maps [13], the neighbors of the winner, $i = k \pm 1$, are also updated according to (6) to preserve topological ordering in the digital coding.

The modification in the update rule (6) is to fix the update amplitude by thresholding:

$$\Delta z_{kj} = \lambda(I_j, z_{kj}) \mathrm{sgn}(I_j - z_{kj}) \tag{11}$$

A constant-amplitude, variable-polarity discrete update is easier to implement than a continuous update, and gives superior results in the presence of analog imprecisions in the implementation [22]. The granular effect of coarse updates is avoided by reducing the update constant β.

Dynamic refresh for long-term analog storage of the weights z_{ij} is achieved using the technique of binary quantization and partial incremental refresh [9]. The technique counteracts drift due to leakage in volatile storage, by maintaining the analog value near one of quantized levels. The stable levels of the dynamic memory are defined by the transition levels of a binary quantization function Q, which maps analog values to binary values $\{-1, 1\}$. When $Q(z_{ij}) = 1$, the analog memory value z_{ij} is slightly decreased, and conversely when $Q(z_{ij}) = -1$, z_{ij} is slightly increased:

$$\Delta z_{ij} = -\mu Q(z_{ij}) \tag{12}$$

Periodic iteration of updates (12) yields long-term stable memory as long as the update amplitude μ is larger than the worst-case drift in between refresh iterations, and significantly smaller than the separation between memory levels [9].

4. VLSI Implementation

The circuits are implemented in current-mode CMOS technology, with MOS transistors operated in subthreshold for low-power dissipation. The use of lateral bipolar transistors offers the advantages of a BiCMOS process while maintaining full compatibility with standard (digital) single-poly CMOS processes.

4.1. Distance Estimation

The circuit diagram of the distance estimation cell is shown in Fig. 2, and the layout of the cell, measuring 43 μm $\times$ 43 μm in 1.2 μm CMOS technology, is given in Fig. 3.

As explained above, the choice function and vigilance measures for Fuzzy ART, as well as the distance metric for VQ, are decomposed in terms of the fuzzy maximum and the minimum of I_j and z_{ij}, accumulated separately onto two wires T_i^+ and T_i^- (9) and combined outside of the array. Their computation is performed by modulating the Early effect (collector conductance) of a matched (double) pair of bipolar transistors Q1–Q4 and Q2–Q3, by means of MOS transistors M1, M2, M3 and M4 connected as source followers. Parallel and series connections in the source followers yield the maximum and minimum of I_j and z_{ij}, respectively, in the output currents.

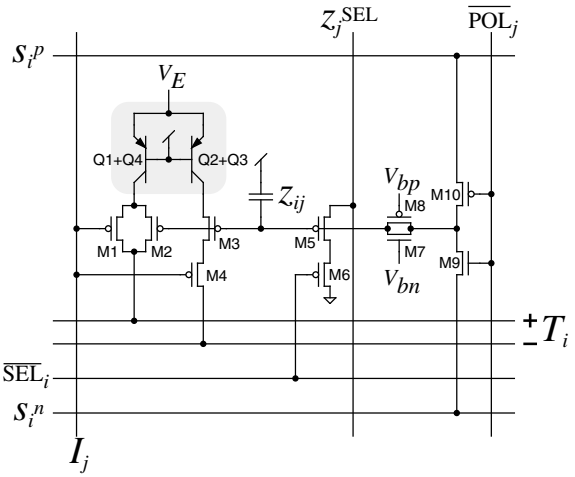

Fig. 2. Circuit schematic of the Fuzzy ART and VQ template matching cell, with integrated learning and template refresh. The dashed inset indicates a matched double pair of lateral bipolar transistors.

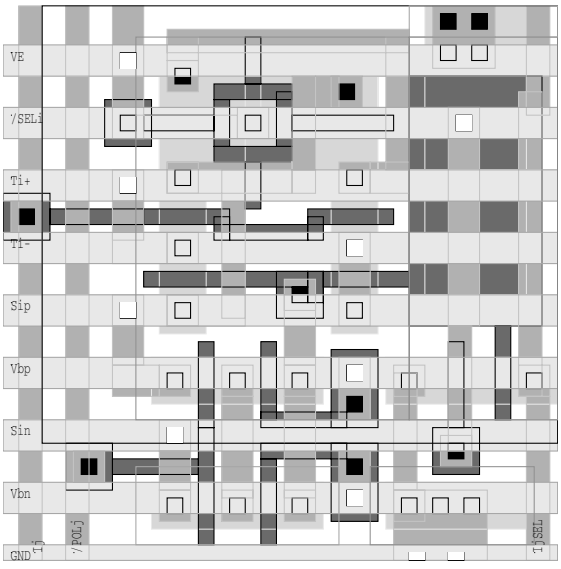

Fig. 3. Layout of the Fuzzy ART and VQ template matching cell, of size 43 μm $\times$ 43 μm in 1.2 μm CMOS technology.

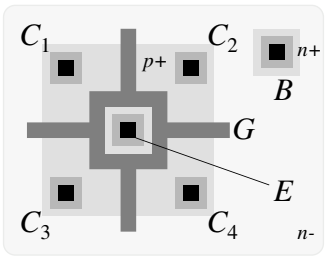

Fig. 4. Centroid geometry of the matched double pair of lateral bipolar transistors, in conventional n-well CMOS technology.

A centroid geometry, shown in Fig. 4, is used for improved matching between the bipolar transistor currents that supply the differential output. By combining collector outputs in pairs $C_1 + C_4$ and $C_2 + C_3$, systematic variations and gradients in geometry are cancelled to first order. Matching is important, since the collector conductance is relatively small. The Early effect is maximized by using a minimum length geometry for the base, equaling the minimum length of an MOS transistor.

The winner-take-all (WTA) is implemented as a variation on the standard current-mode design in [10,11] with the addition of a triggered voltage-mode output stage for improved settling accuracy and speed [17].

4.2. Learning and Refresh

Transistors M5 through M10 implement the incremental update Δz_{ij}, when the cell is selected either for refresh, or for learning (when $k^{WTA} \equiv i$). The polarity $\overline{\mathrm{POL}}_i$ of the fixed-amplitude update Δz_{ij} is precisely implemented by means of a binary controlled charge pump [9]. The charge pump is free of switch charge injection parasitics, by avoiding clock signals on

the MOS gates that couple capacitively into the storage capacitor. V_{bn} and V_{bp} are biased deep in subthreshold for precisely controlled increments and decrements as small as 10 μV. The timing of the update (and selection of the template) is performed by means of signals S_n^j and S_p^j [22].

The update polarity is computed externally to the array, by circuitry common for all cells on the same column, shown at the top of Fig. 1. This arrangement is most space efficient since only one row of cells needs to be updated at once. A global signal (LEARN) selects the mode of operation, learning or refresh.

Figure 5 shows the simplified schematic of the external learning cell, one per column of VQ distance cells. The circuit receives the selected analog template value z_{ij} on the line Z_j^{SEL} along with the input I_j which are used to generate the update polarity $\overline{\text{POL}}_i$ and supply it to the selected distance cell. When a distance cell is selected, switch M6 is closed and transistors M5–M6 along with M11–M13 implement a comparator. The update is performed in the cell according to $\overline{\text{POL}}_i$ by activating the signals S_n^j and S_p^j for the entire row of selected cells.

The selection of $\lambda(I_j, z_{ij})$ in (7) is implicit in the computed polarity of the update in (11), and is automatically enforced by limiting updates to negative polarities only. This is implemented by pulling V_{bn} in Fig. 2 to zero whenever LEARN is active, in Fuzzy ART mode. The remaining V_{bp} pMOS tail current then provides the negative β updates. Note that negative z_{ij} updates correspond to positive charge injection on the storage capacitor, since the output currents on the T_i^+ and T_i^- lines decrease when the stored voltage increases.

In learning mode (LEARN≡1), the polarity $\overline{\text{POL}}_i$ is computed by comparing z_{kj} with the input I_j, yielding a fixed-size update according to (11). In refresh mode (LEARN≡0), z_{kj} is compared with an external reference signal $z_{\text{D/A}}$ to construct the binary quantization function used for partial incremental refresh [9] in (12). As in [23], the binary quantization Q of z_{kj} is obtained by retaining the least significant bit (LSB) of analog-to-digital (A/D) conversion of z_{kj}. A single-slope sequential A/D is implemented for simplicity, using a D/A signal on $z_{\text{D/A}}$, ramped up in discrete steps synchronously with the alternating LSB. When the comparator flips sign, the instantaneous LSB value is sampled and latched to generate the update polarity $Q(z_{kj})$, producing an update according to (12).

5. Experimental Results

The layout of the 8×16 learning Fuzzy ART classifier and vector quantizer, implemented in 1.2 μm CMOS technology, is shown in Fig. 6. The experimental results described here have been obtained from this chip, and are limited to Fuzzy ART operation. A previous

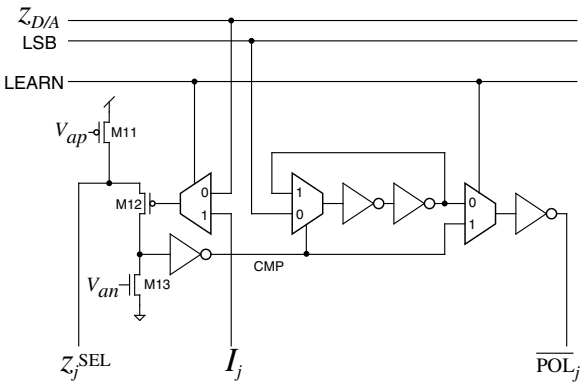

Fig. 5. Simplified schematic of the learning and refresh circuitry, in common for a column of Fuzzy ART/VQ cells. Analog multiplexers are implemented with complementary CMOS switches.

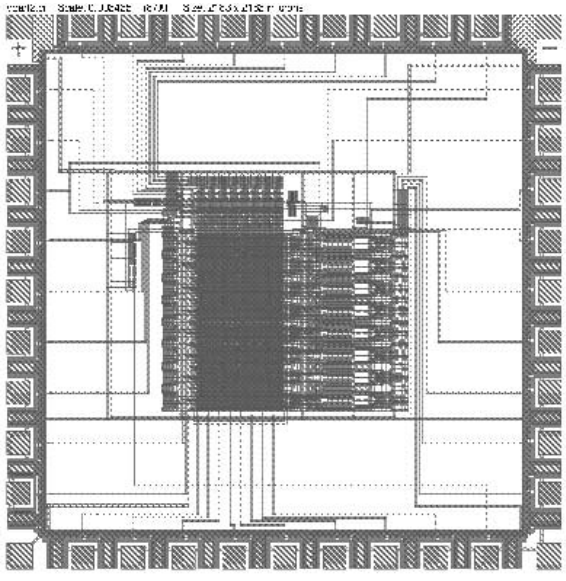

Fig. 6. Layout of the 8×16 array, analog learning Fuzzy ART classifier and VQ. The die size is 2.2×2.25 mm^2 in 1.2 μm CMOS technology.

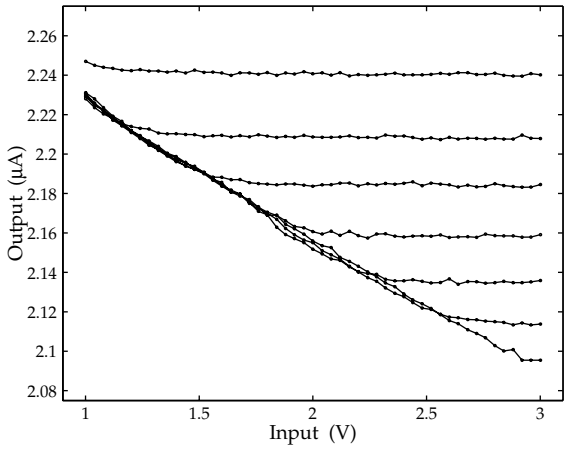

Fig. 7. Measured Fuzzy ART choice function for one row of cells, sweeping one input component while fixing the other 7 inputs.

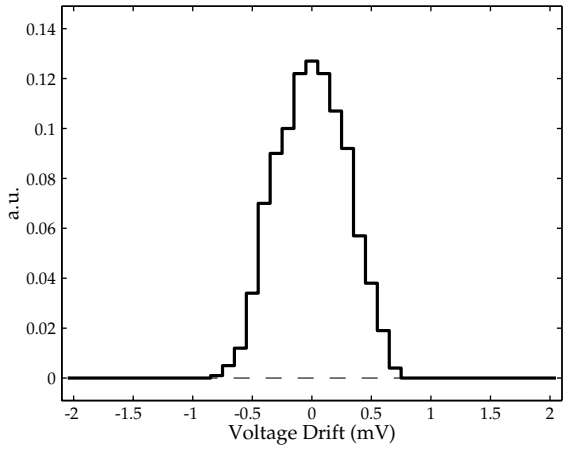

Fig. 9. Measured spread of the drift over time of the stored analog voltage during refresh.

16×16 version in 2 μm CMOS technology, with experimental results in VQ mode, are described in [7] and [8].

The fuzzy min distance metric is illustrated in Fig. 7, obtained by sweeping one of the 8 inputs while fixing the other inputs to the template values. Note the reverse polarity of the input needed for a proper "min" operator; this is inconsequential since inputs are complement encoded and so both polarities are presented together.

Results for dynamic refresh of the templates at 128-level quantization are shown in Fig. 8, obtained by observing the drift in stored voltage level on one of the cells over 100 refresh cycles (several minutes), for different initial values of the voltage z_{ij}. The corresponding drift without refresh would have been several volts. Further improvements in resolution and stability can be achieved, at some expense in silicon area, by using the A/D/A quantizer in [23]. Stable long-term storage at 256-level quantization, in excess of 10^9 refresh cycles (more than a week), was previously demonstrated with this technique in [24].

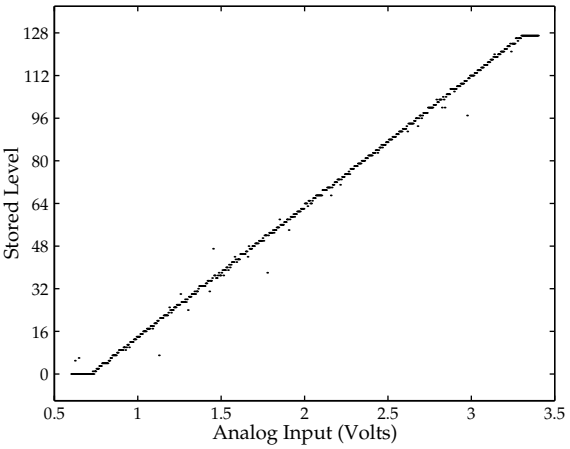

Fig. 8. Stability of the analog memory array, at 128-level quantization. Measured drift over 100 self-refresh cycles (several minutes), from different initial values.

The spread of the stored analog voltage, during refresh over time, is recorded in Fig. 9. With an update amplitude of 0.4 mV, the excursion of the voltage is limited to about 1 mV. An external refresh scheme with off-chip memory would amount to the same, or larger, excursion in voltage during refresh. Since the on-chip refresh can proceed in the background, without interrupting the fuzzy ART or VQ computation, the dynamic analog memory is transparent to the user.

Learning tests which validate the functionality of the classifier in LVQ and Fuzzy ART learning modes, with adaptive updates according to (11), are illustrated in Fig. 10. While the LVQ updates are symmetric in upward and downward directions, the asymmetry in charging rate for upward and downward transitions is a feature of Fuzzy ART—stability of neural plasticity in the presence of changing and noisy input conditions [3].

Applications of the chip, and its scalable extensions, to speech and image coding are the subject of continued research. We are currently also investigating connections between adaptive classifiers of this type, and a broader class of kernel "machines" for on-line incremental supervised learning [25], and their VLSI implementation [26].

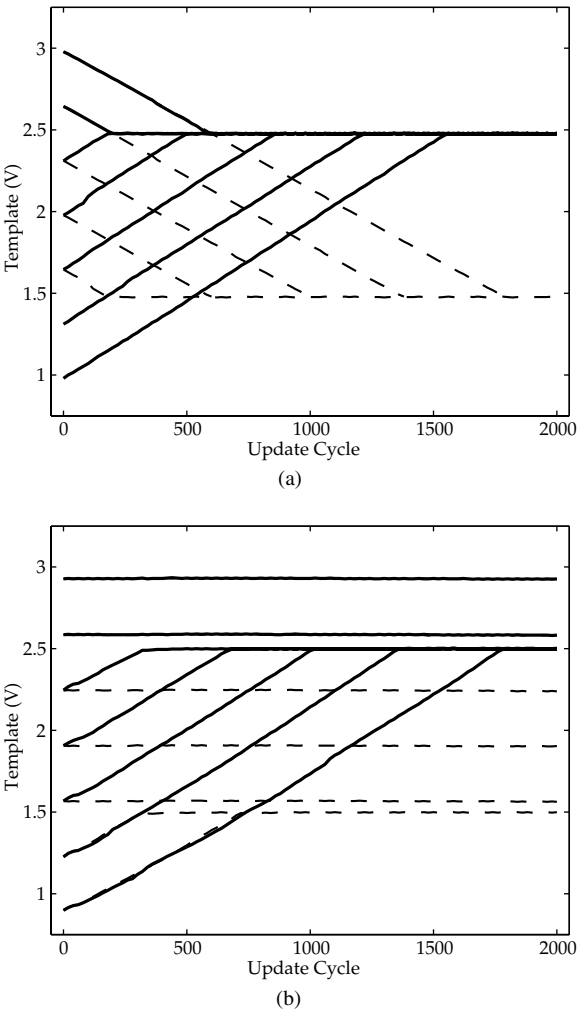

Fig. 10. Template adaptation recorded from a single cell in learning mode, under fixed input, from a range of initial template values. (a) Learning vector quantization (and Fuzzy ART fast learning) mode. (b) Fuzzy ART learning mode. Solid lines: 2.5 V fixed input; and dotted lines: 1.5 V fixed input.

6. Conclusions

We have implemented an asynchronous mixed-mode CMOS VLSI system capable of classifying and learning in real-time. We presented a parallel architecture and corresponding analog VLSI BiCMOS implementation of a fuzzy ART classifier, fabricated in a single-poly CMOS process using lateral bipolar transistors. The chip can be configured for variants on fuzzy ART such as VQ and Kohonen self-organizing maps. It incorporates analog storage of the templates, sharing the same circuitry used for learning. With a dense cell size of 71×71 λ units in scalable MOSIS technology, the integration of a 256-input, 1024-category classifier is feasible on a 1 cm^2 die in a 0.35 μm CMOS process ($\lambda = 0.2\,\mu$m). Extension to full Fuzzy ARTMAP capability (for pattern association) would reduce the density roughly by a factor two, still supporting 256 inputs and 512 categories.

Appendix: Simplifications in Implemented Fuzzy ART Choice Function

For complement-encoded inputs, we can expand the fuzzy-min operator as

$$\begin{aligned} |\mathbf{I} \wedge \mathbf{z}_i| &= \sum_i (\min(I_j, z_{ij}) + \min(\overline{I_j}, z'_{ij})) \\ &= N I_{\max} + \sum_i (\min(I_j, z_{ij}) \\ &\quad - \max(I_j, \overline{z'_{ij}})) \end{aligned} \tag{A.1}$$

where $\overline{z'_{ij}} = I_{\max} - z'_{ij}$ is the complement of the weight corresponding to the complement-encoded input. This and similar expressions for $|\mathbf{z}_i|$ allow to expand the choice function as

$$\begin{aligned} T_i &= \frac{|\mathbf{I} \wedge \mathbf{z}_i|}{\alpha + |\mathbf{z}_i|} \\ &\approx \frac{1 + \frac{\sum_i (\min(I_j, z_{ij}) - \max(I_j, \overline{z'_{ij}}))}{N I_{\max}}}{1 + \frac{\sum_i (z_{ij} - \overline{z'_{ij}})}{N I_{\max}}} \end{aligned} \tag{A.2}$$

$$\begin{aligned} &\approx 1 + \frac{\sum_i (\min(I_j, z_{ij}) - \max(I_j, \overline{z'_{ij}}))}{N I_{\max}} \\ &\quad - \frac{z_{ij} - \overline{z'_{ij}}}{N I_{\max}} \end{aligned} \tag{A.3}$$

assuming $N I_{\max}$ is much larger than other terms in the expression. This approximation is almost perfect near 'resonance,' and differences between exact and approximated versions are significant only when the distance between input and template is large, and the vigilance condition is less likely to be met. A more precise approximation follows.

First, we further simplify the expression, and eliminate the need of computing and subtracting the summed weights $|\mathbf{z}_i|$ altogether. Since at initial coding $z_{ij} \equiv \overline{z'_{ij}}$, and since the weights z_{ij} and z'_{ij} only decrease under

the learning updates, it follows that $z_{ij} \leq \overline{z'_{ij}}$. Thus,

$$\begin{aligned}&\min(I_j, z_{ij}) - \max(I_j, \overline{z'_{ij}}) - z_{ij} + \overline{z'_{ij}} \\ &\quad = \min(I_j, \overline{z'_{ij}}) - \max(I_j, z_{ij}) \end{aligned} \tag{A.4}$$

and the choice function reduces to

$$\begin{aligned}T_i &= \frac{N I_{\max} - \sum_i (\max(I_j, z_{ij}) - \min(I_j, \overline{z'_{ij}}))}{N I_{\max}} \\ &= \frac{2N I_{\max} - \sum_i (\max(I_j, z_{ij}) + \max(\overline{I_j}, z'_{ij}))}{N I_{\max}} \\ &= 2 - \frac{|\mathbf{I} \vee \mathbf{z}_i|}{N I_{\max}} \end{aligned} \tag{A.5}$$

In other words, maximizing the fuzzy min choice function normalized by the weights is, approximately, equivalent to minimizing a modified fuzzy max choice function, without weight normalization.

It can be verified that this new choice function for the Fuzzy-ART algorithm produces valid clustering behavior, in the sense that it satisfies the properties outlined in [3]. However, for correct Fuzzy-ARTMAP operation, it is not possible to neglect α in the approximation of the choice function (1), at the risk of invalidating certain properties such as "Direct Access to Subset and Superset Patterns" [4]. Nevertheless, a non-zero value for α is easily accounted for in the above approximations, changing (A.5) into

$$\begin{aligned}T_i &\approx \frac{1 + \frac{\sum_i (\min(I_j, z_{ij}) - \max(I_j, \overline{z'_{ij}}))}{N I_{\max}}}{1 + \frac{\sum_i (z_{ij} - \overline{z'_{ij}})}{N I_{\max} + \alpha}} \\ &\approx 1 + \frac{\sum_i (\min(I_j, z_{ij}) - \max(I_j, \overline{z'_{ij}}))}{N I_{\max}} \\ &\quad - (1 - \alpha')(z_{ij} + \overline{z'_{ij}}) N I_{\max} \\ &= 2 - \alpha' - \frac{(1 - \alpha')|\mathbf{I} \vee \mathbf{z}_i| - \alpha' |\mathbf{I} \wedge \mathbf{z}_i|}{N I_{\max}} \end{aligned} \tag{A.6}$$

where $\alpha' = \alpha / N I_{\max}$. Thus, for Fuzzy ARTMAP with nonzero value for α, a valid choice function can still be constructed by linearly combining fuzzy max and min operations.

Acknowledgment

This work was supported by ARPA/ONR MURI N00014-95-1-0409, by ONR YIP N00014-99-1-0612, and by NSF Career MIP-9702346. Chip fabrication was provided through MOSIS.

References

1. Grossberg, S., "Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors." *Biological Cybernetics* 23, pp. 121–134, 1976.
2. Gersho, A. and Gray, R. M., *Vector Quantization and Signal Compression.* Kluwer Academic, Norwell, MA, 1992.
3. Carpenter, G. A., Grossberg, S. and Rosen, D. B., "Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system." *Neural Networks* 4, pp. 759–771, 1991.
4. Serrano, T., Linares, B. and Andreou, A., *Adaptive Resonance Theory Microchips.* Kluwer Academic, Norwell MA, 1998.
5. Cauwenberghs, G. and Bayoumi, M. (eds.), *Learning on Silicon.* Kluwer Academic, Norwell MA, 1999.
6. Cohen, M., Abshire, P. and Cauwenberghs, G., "Mixed-mode VLSI implementation of fuzzy ART," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS'98),* Monterey, CA, May 31–June 3, 1998.
7. Lubkin, J. and Cauwenberghs, G., "A micropower learning vector quantizer for parallel analog-to-digital data compression," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS'98).* Monterey, CA, May 31–June 3, 1998.
8. Lubkin, J. and Cauwenberghs, G., "A learning parallel analog-to-digital vector quantizer." *Journal of Circuits, Systems and Computers* (Special Issue on Analog and Digital Arrays) 8(5–6), pp. 605–614, 1998.
9. Cauwenberghs, G. and Yariv, A., "Fault-tolerant dynamic multilevel storage in analog VLSI." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 41(12), pp. 827–829, 1994.
10. Lazzaro, J., Ryckebusch, S., Mahowald, M. A. and Mead, C. A., "Winner-take-all networks of O(n) complexity," *Advances in Neural Information Processing Systems,* Morgan Kaufman, San Mateo, CA, 1989, vol. 1, pp. 703–711.
11. Andreou, A. G., Boahen, K. A., Pouliquen, P. O., Pavasovic, A., Jenkins, R. E. and Strohbehn, K., "Current-mode subthreshold MOS circuits for analog VLSI neural systems." *IEEE Transactions on Neural Networks,* 2(2), pp. 205–213, 1991.
12. Serrano-Gotarredona, T., Linares-Barranco, B. and Huertas, J. L., "A real time clustering CMOS neural engine." In: D. S. Touretzky (ed.), *Advances in Neural Information Processing Systems.* Morgan Kauffmann, San Mateo, CA, 1995, vol. 7.
13. Hochet, B., Peiris, V., Abdot, S. and Declercq, M. J., "Implementation of a learning Kohonen neuron based on a new multilevel storage technique." *IEEE J. Solid-State Circuits* 26, pp. 262–267, 1991.
14. Fang, W. C., Sheu, B. J., Chen, O. T. C. and Choi, J., "A VLSI neural processor for image data-compression using self-organization networks." *IEEE Transactions on Neural Networks,* 3(3), pp. 506–518, 1992.
15. He, Y. and Cilingiroglu, U., "A charge-based on-chip adaptation Kohonen neural network." *IEEE Transactions on Neural Networks* 4(3), pp. 462–469, 1993.
16. Tuttle, G. T., Fallahi, S. and Abidi, A. A., "An 8b CMOS vector A/D converter." *ISSCC Technical Digest.* IEEE Press, 36, pp. 38–39, 1993.
17. Cauwenberghs, G. and Pedroni, V., "A charge-based CMOS parallel analog vector quantizer," *Advances in Neural Information*

Processing Systems. MIT Press, Cambridge, MA, 1995, vol. 7, pp. 779–786.
18. Konda, M., Shibata, T., Ohmi, T., "Neuron-MOS correlator based on manhattan distance computation for event recognition hardware." *Dig. International Symposium on Circuits and Systems.* Atlanta, GA, 1996.
19. Wang, J. W., Coggins, R. J. and Jabri, M. A., "Micropwer analogue building blocks for template matching in implantable devices," in *Proc. 8th Australian Conf. Artificial Neural Networks,* pp. 177–180, 1997.
20. Granger, E., Blaquière, Y., Savaria, Y., Cantin, M.-A. and Lavoie, P., "A VLSI architecture for fast clustering with fuzzy ART neural networks." *J. Microelectronic Systems Integration* 5(1), pp. 3–18, 1997.
21. Coggins, R. J., Wang, R. J. W. and Jabri, M. A., "A micropower adaptive linear transform vector quantiser." In: G. Cauwenberghs and M. Bayoumi (eds.), *Learning on Silicon*. Norwell Kluwer Academic, MA, 1999.
22. Cauwenberghs, G., Analog VLSI stochastic perturbative learning architectures." *Int. J. Analog Integrated Circuits and Signal Processing* 13(1/2), pp. 195–209, 1997.
23. Cauwenberghs, G., "A micropower CMOS algorithmic A/D/A converter." *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 42(11), pp. 913–919, 1995.
24. Cauwenberghs, G., "Analog VLSI long-term dynamic storage," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'96),* Atlanta, GA, 1996, vol. III, pp. 334–337.
25. Cauwenberghs, G. and Poggio, T., "Incremental and decremental support vector machine learning," submitted (also available as MIT AI memo, http//www.ai.mit.edu), 2000.
26. Genov, R. and Cauwenberghs, G., "Charge-mode parallel architecture for matrix-vector multiplication," in *Proc. 43rd IEEE Midwest Symp. Circuits and Systems (MWSCAS'2000).* Lansing, MI, August 8–11, 2000.

Jeremy Lubkin received the B.S. and M.S. degrees in electrical engineering from the Johns Hopkins University in 1997 and 1998, respectively. He is currently employed at Tality, Columbia, MD, where he is active in the design of analog and mixed-signal CMOS and BiCMOS integrated systems.

Gert Cauwenberghs received the Engineer's degree in Applied Physics from the University of Brussels, Belgium, in 1988, and the M.S. and Ph.D. degrees in electrical engineering from the California Institute of Technology in 1989 and 1994. In 1994, he joined Johns Hopkins University where he is now associate professor of Electrical and Computer Engineering. During 1998–1999 he was on sabbatical as visiting professor of Brain and Cognitive Science at the Center for Computational and Biological Learning, Massachusetts Institute of Technology, and at the Center for Adaptive Systems, Boston University. His research covers VLSI circuits, systems and algorithms for parallel signal processing, adaptive neural computation, and low-power coding and instrumentation. He has organized special sessions at conferences and journal special issues on learning, adaptation and memory, and recently co-edited a book on *Learning on Silicon* (Kluwer, 1999). He was Francqui Fellow of the Belgian American Educational Foundation in 1988, and received the National Science Foundation Career Award in 1997, the Office of Naval Research Young Investigator Award in 1999, and the Presidential Early Career Award for Scientists and Engineers (Pecase) in 2000. He is associate editor of the *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* and Chair of the IEEE Circuits and Systems Society Technical Committee on Analog Signal Processing.