CNNA-94 Third IEEE International Workshop on Cellular Neural Networks and their Applications
Rome, Italy, December 18-21, 1994

Hardware-Oriented Learning for Cellular Neural Networks

Andreas J. Schuler, Martin Brabec, Dirk Schubel, and Josef A. Nossek
Institute for Network Theory and Circuit Design
Technical University of Munich, Germany
Email: ansc@nws.e-technik.tu-muenchen.de

Abstract *This paper presents an approach to learning, which focusses on finding a set of parameter values taking into account the nonidealities of a specific implementation. Therefore learning is done on a more accurate model of a CMOS cell, and not on the original CNN model proposed in 1988 [1, 5].*

This hardware-oriented approach will be applied to a current-mode CNN-model based on the full-signal-range model published in [10, 2], where the dynamic block consists of two current mirrors. It is shown, that a two-quadrant multiplier is sufficient for the multiplication with the template coefficients, by changing the model, further reducing the area consumption.

Using a hardware-oriented approach to learning thus not only allows to find template values for a specific VLSI-implementation, but may also lead to further simplifications of CNN-implementations.

1 Introduction

Since their introduction in 1988 [1] the design and learning of cellular neural networks (CNNs) has been an interesting research topic. On the other hand VLSI-implementation of CNNs is a crucial point for the application of CNNs to real world problems. Currently the hardware implementation persuits two main directions, namely the development of an analogic CNN Universal Machine [11], and the efficient implementation of simple, specialized CNN devices with optimized area consumption [10, 14].

For the second approach modifications of the original model have proven to be useful, namely the use of different nonlinear functions and the full signal range, leading to a simplified VLSI-implementation.

To find appropriate parameter values design or learning should be done on the modified model, taking into account the nonidealities of VLSI-implementations, in order to guarantee the correct operation of the actual chip.

In this paper, an approach for hardware-oriented learning is presented which finds the parameters of a simplified full signal range CNN using a ODE-model of the basic building blocks for current-mode CNNs. First, a simplified full-signal-range CNN architecture is presented, which requires only two-quadrant multipliers and thereby reduces the area consumption; then, the

0-7803-2070-0/94/$4.00 © 1994 IEEE

hardware-oriented ODE-model of the basic building blocks will be shown. Backpropagation-through-Time [9, 7] can be used to learn the parameters of this ODE-model.

2 A Simplified Full-Signal-Range CNN (FSR-CNN)

The full signal range CNN (FSR-CNN) model presented in [10] originates from the fact, that the dynamic operator of a FSR-CNN can be easily implemented by a current-mode lossy integrator. The signal range of the state variables is equal to the signal range of the output, and therefore the cell output and cell state can be merged into a single variable.

In this paper a modified FSR-CNN, with a signal range of $[0, 1]$ is proposed, given by

$$\tau \dot{x}_c = \begin{cases} 0 & \text{if} \quad x_c = 0 \\ -x_c + \sum\limits_{d \in \mathcal{N}_c} a_{d-c} x_d + k_c & \text{if} \quad 0 < x_c < 1 \\ 0 & \text{if} \quad x_c = 1 \end{cases} \tag{1}$$

The modified and the original FSR-CNN are equivalent since the signals and weights are related by a linear transformation,

$$x_c = \frac{1}{2}(\tilde{x}_c + 1); \qquad \dot{x}_c = \frac{1}{2}\dot{\tilde{x}}_c \tag{2}$$

$$\tau = \tilde{\tau}; \qquad a_{d-c} = \tilde{a}_{d-c}; \qquad k_c = \frac{1}{2}\left(\tilde{k}_c - \sum_{d \in \mathcal{N}_c} a_{d-c} + 1\right). \tag{3}$$

The variables $\tilde{x}_c$, $\tilde{\tau}$, $\tilde{a}_{d-c}$ and $\tilde{k}_c$ are the corresponding variables of the original FSR-CNN, which has a signal range of $[-1, +1]$. From (3) it can be seen, that the template coefficients remain the same, and only the bias k_c has to be changed to obtain the same behaviour.

3 Architecture of the modified FSR-CNN

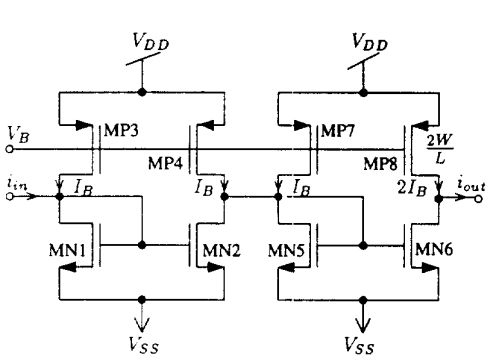
The FSR-CNN behaves similar to the original model by Chua and Yang, i.e. the equilibrium points are corresponding. Moreover it is much simpler to implement in current mode techniques. The two kinds of building blocks necessary for implementation are the static blocks for summation, multiplication and nonlinearity, and the dynamic block.

The dynamic operator, shown in Fig. 3a, consists of two cascaded bias shifted current mirrors only. Dynamic operation is achieved due to the parasitic gate to source capacitances. Due to the nonlinear behaviour of the MOSFET transistors the necessary nonlinearity is achieved at the same time. The linear transformation of the signal range of the output current i_{out} to positive values is obtained by bias-shifting the output of the second current mirror. A detailed ODE model of the dynamic block will be derived in the next section.

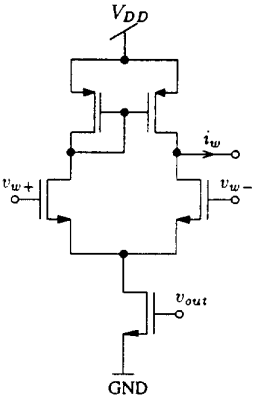
In contrast to the original FSR-CNN model our modified model possesses only positive states, and as a result only two-quadrant multipliers are required for the multiplication of the weights a_{d-c} and the states x_d. The two-quadrant multiplier can be implemented using a differential pair (Fig. 3b). The weight signal is differential, and can be defined as $v_w = v_{w+} - v_{w-}$. The output of the multiplier, i.e. the weighted signal i_w, is in current form. But the input v_{out} is given in form of a voltage. The positive output current of the dynamic block is converted to a positive voltage using a simple linear I-V converter.

For the summing block a current-input current-output adder is realized by Kirchhoffs Current Law at the input node of the dynamic block. Additionally to the weighted current signals of surrounding cells a bias current must be added according to (3).

In an actual CMOS implementation of course, no truely static blocks exist, but the transient behaviour of the static blocks is much faster than the time constant of the dynamic block τ.



(a) Dynamic current-mode block



(b) Two-quadrant multiplier

Figure 1: Two basic current-mode building blocks of the modified FSR-CNN. (a) Dynamic block. (b) Two-quadrant multiplier

4 A Hardware-oriented ODE-Model

For gradient-based learning a mathematical description of the static and dynamic blocks of an implemented cell is necessary. With a suitable formulation parameter values can be found by learning, which guarantee the functionality of the actual realization.

First, we will focus on the dynamic model of a biased current mirror. Then the ordinary differential equations of the dynamic block will be derived. Together with the description of the multipliers, which are assumed to be static, a hardware-oriented description of the modified FSR-CNN is achieved.

4.1 Dynamic Model of the Biased Current Mirror

The two biased current mirrors forming the dynamic block differ only in the bias of the second MOSFET, which is I_B for the first, and $2I_B$ for the second. In a first step we will calculate the differential equation for one current mirror shown in Fig. 2.

From the I-V characteristic of the MOSFET [3, 4], the capacitive behaviour of the gate node, and Kirchhoffs current law, the following equations can be obtained as long as $i_{in1} > -I_B$:

$$\begin{aligned} i_{d1} &= k\,(1+\lambda v_1)\,(v_1 - v_T)^2 \quad , \quad i_{g1} = C\dot{v}_1 \\ i_{d2} &= k\,(1+\lambda v_2)\,(v_1 - v_T)^2 \quad , \quad i_{g2} = C\dot{v}_1 \\ i_{in1} + I_B &= i_{d1} + i_{g1} + i_{g2} \quad , \quad I_B - i_{out1} = i_{d2}. \end{aligned} \tag{4}$$

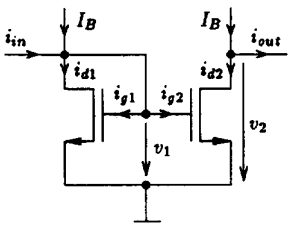


Figure 2: Biased current mirror

From this equations, we find that $i_{out1} < I_B$ and

$$v_1 = \sqrt{\frac{I_B - i_{out1}}{k\,(1+\lambda v_2)}} + v_T\,. \tag{5}$$

After the differentiation v_1 and $\dot{v}_1$ can be eliminated to obtain the differential equation

$$\begin{aligned} i_{in1} + I_B \;=\; & \left(1 + \lambda\sqrt{\frac{I_B - i_{out1}}{k\,(1+\lambda v_2)}}\lambda v_T\right)\frac{I_B - i_{out1}}{1+\lambda v_2} + \\ & + C\left[\frac{-i_{out1}}{\sqrt{k\,(1+\lambda v_2)\,(I_B - i_{out1})}} - \frac{2k\lambda\sqrt{I_B - i_{out1}}\,\dot{v}_2}{(k\,(1+\lambda v_2))^{3/2}}\right]. \end{aligned} \tag{6}$$

Neglecting channel length modulation, i.e. $\lambda = 0$, (6) can be simplified

$$i_{out1} = -\frac{1}{C}\sqrt{k\,(I_B - i_{out1})}\,(i_{out1} + i_{in1})\,. \tag{7}$$

For $i_{in1} < -I_B$, it can be seen that $i_{out1} = I_B$ and $\dot{i}_{out1} = 0$. The dynamic of (7) guarantees, that $i_{out1} < I_B$. Thus the dynamical behaviour of the biased current mirror is described by the nonlinear differential equation (7).

4.2 Differential Equation of the Dynamic Block

The equation of the second mirror can be obtained similarly, where the signal range of i_{out2} is restricted to $i_{out2} < 2I_B$:

$$i_{out2} = -\frac{1}{C}\sqrt{k\,(2I_B - i_{out2})}\,(-i_{out2} - i_{in2} + I_B)\,, \tag{8}$$

as long as $i_{in2} > -I_B$. For $i_{in2} < -I_B$, we find $i_{out2} = 2I_B$, $\dot{i}_{out2} = 0$.

Putting (7) and (8) together and with $i_{in2} = i_{out1}$, the dynamical behaviour of one cell is described by two nonlinear differential equations. The output current of the dynamic block $i_{out} = i_{out2}$ possesses a signal range $0 < i_{out} < 2I_B$.

4.3 Initial Conditions

For the initialization the cells are disconnected from each other, and each cell is loaded with $i^\circ_{in1,c}$. From (7) and (8) it can be seen that after a short transient: $i^\circ_{out1,c} = -i^\circ_{in1,c}$, $i^\circ_{out2,c} = -i^\circ_{in2,c} + I_B = i^\circ_{in1,c} + I_B$.

5 Learning on the Hardware-Oriented ODE-Model

In the previous section, we have found, that each cell can be modelled by two nonlinear differential equations (7, 8). The proposed hardware-oriented ODE-model of our improved FSR-CNN consisting of N cells is given by a set of $2N$ differential equations.

These $2N$ state variables can be arranged into a single state vector $\mathbf{x}$:

$$\mathbf{x} = (z_{out1,1}, z_{out2,1}, \cdots, z_{out1,N}, z_{out2,N})^{\mathsf{T}} , \tag{9}$$

and the dynamical system can be written in the common way:

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{p}) . \tag{10}$$

The parameter vector $\mathbf{p}$ contains all controllable variables of the model, most importantly the voltages for the weights, but may as well contain physical parameters like the channel width and length of the MOSFETs.

A common way for learning in neural networks is to define an error measure or cost function of the fixed points and the desired outputs (*Recurrent Backpropagation* [8]) or in general of the trajectory of the system and the desired trajectory (*Backpropagation-Trough-Time* [7]).

Modified versions of recurrent backpropagation and backpropagation-through-time have been developed [13, 12] to make sure, that the CNN will not only have the desired fixed point, but evolve from a given initial condition into the corresponding fixed point.

For each state variable an error $E_c(\mathbf{p})$ is defined using suitable functions $e_{1,c}$ and $e_{2,c}$ and the desired value of the variable d_c. The total error for a particular problem is the sum of the individual errors of all states and all learning examples. The gradient of this error with respect to the weights, which can be used to descend to a local minimum of the error, can be simplified

$$E_c(\mathbf{p}) = e_{1,c}(x_c(T)d_c) \int_0^T e_{2,c}(x_c(t)d_c)dt , \qquad \frac{\partial E_c}{\partial \mathbf{p}} = \int_0^T \lambda_c \frac{\partial F_c}{\partial \mathbf{p}}\, dt , \tag{11}$$

using the associated dynamical system

$$\dot{\lambda}_c = \lambda_c - \sum_{d \in \mathcal{N}_c} \frac{\partial F_d}{\partial x_c} \lambda_d - e_{1,c}(x_c(T)d_c) e'_{2,c}(x_c(t)d_c)d_c , \tag{12}$$

which has to be integrated backward in time, since the boundary value of λ_c is known at the terminal time T:

$$\lambda_c(T) = e'_{1,c}(x_c(T)d_c)d_c \int_0^T e_{2,c}(x_c(t)d_c)dt . \tag{13}$$

The partial derivatives of the right hand side of the dynamical system (10), $\partial F_d/\partial x_c$ and $\partial F_c/\partial \mathbf{p}$ can be obtained from the state equations of the ODE model (7, 8).

The backward integration requires the storage of the whole trajectory of the state variables. This limits the applicability to small problems. However, because of the local processing in CNNs, learning with small examples is sufficient, and therefore this is not a serious restricion.

6 Conclusion

Learning can not only be used to find new template values for the original CNN model. But hardware oriented learning provides a tool for finding template values for an actual VLSI-implementation. The properties and nonidealities of the circuit can be taken into account, thus the correct functional behaviour can be trained. Although the ODE-description of the actual MOS circuit can become quite complicated, especially as more effects like channel length modulation, finite I/O resistances, and finite frequency response of the static blocks are considered, gradient based learning still can be applied. This could lead to even further simpler cells, allowing larger cell densities with better yield concerning functional behaviour.

References

[1] L. O. Chua and Lin Yang. Cellular Neural Networks: Theory. *IEEE Transactions of Circuits and Systems*, 35:1257–1272, 1988.

[2] Servando Espejo. *Redes Neuronales Celulares: Modelado Y Diseño Monolítico*. PhD thesis, Universidad De Sevilla, Spain, 1994.

[3] Paul R. Gray and Robert G. Meyer. *Analysis and Design of Analog Integrated Circuits*. John Wiley & Sons, Inc., New York, 3 edition, 1993.

[4] Roubik Gregorian and Gabor C. Temes. *Analog MOS Integrated Circuits*. John Wiley & Sons, Inc., New York, 1986.

[5] J. A. Nossek, G. Seiler, T. Roska, and L. O. Chua. Cellular Neural Networks: Theory and circuit design. Technical Report TUM-LNS-TR-90-7, Technical University Munich, 12 December 1990.

[6] J.A. Nossek, H. Magnussen, P. Nachbar, and A.J. Schuler. Learning algorithms for Cellular Neural Networks. In *Proc. of the International Symposium on Nonlinear Theory and its Applications*, volume 1, pages 11–16, Hawaii, 1993. IEEE.

[7] Barak A. Pearlmutter. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1:263–269, 1989.

[8] Fernando J. Pineda. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, 59(19):2229–2232, November 1987.

[9] Fernando J. Pineda. Recurrent backpropagation and the dynamical approach to adaptive neural computation. *Neural Computation*, 1:161–172, 1989.

[10] Angel Rodríguez-Vaázquez, Servando Espejo, Rafael Domínguez-Castro, Jose L. Huertas, and E. Sánchez-Sinencio. Current-mode techniques for the implementation of continuous- and discrete-time Cellular Neural Networks. *IEEE Transactions of Circuits and Systems, Special Issue on Cellular Neural Networks*, 40(3):132–146, March 1993.

[11] T. Roska and L. O. Chua. The CNN universal machine: An analogic array computer. *IEEE Transactions of Circuits and Systems, Special Issue on Cellular Neural Networks*, 40(3):163–173, March 1993.

[12] A. J. Schuler, P. Nachbar, and J. A. Nossek. State-based backpropagation-through-time for CNNs. In *Proc. of the 11th European Conference on Circuit Theory and Design, Davos, Switzerland*, pages 33–38, 1993.

[13] A. J. Schuler, P. Nachbar, J. A. Nossek, and L. O. Chua. Learning state space trajectories in Cellular Neural Networks. In *Proc. of the second IEEE Int. Workshop on Cellular Neural Networks and their Applications, CNNA-92, Munich, Germany*, pages 68–73, 1992.

[14] Joseph E. Varrientos, Edgar Sánchez-Sinencio, and Jaime Ramírez-Angulo. A current-mode Cellular Neural Network implementation. *IEEE Transactions of Circuits and Systems, Special Issue on Cellular Neural Networks*, 40(3):147–155, March 1993.