Controlled accuracy approximation of the sigmoid function for efficient FPGA-based implementation of artificial neurons

I. del Campo, R. Finker, J. Echanobe and K. Basterretxea

This work proposes a controlled accuracy approximation scheme of the sigmoid function for artificial neuron implementation based on Taylor's Theorem and the Lagrange form of the error. The main advantages of the proposed solution are two: it provides a systematic way to guarantee the required accuracy and it reuses the circuitry of the linear part of the neuron to compute the sigmoid function. The sigmoid derivative is also available for artificial neural networks with online learning capabilities.

Introduction: The number and variety of engineering applications of artificial neural networks (ANNs) have been increasing, ranging from consumer products to industrial process control. Efficient hardware implementations have been developed for numerous applications demanding high performance real-time processing [1]. Hardware implementations of many ANNs involve the computation of a nonlinear activation function. The sigmoid function is one of the most widely used, mainly due to its differentiable nature which makes it suitable for online training. However, this function is costly to implement in digital hardware because it requires the calculation of an exponentiation and a division. To avoid this problem, a number of approximation techniques have been proposed over the years. The most commonly used are look-up tables (LUTs), bit-level mapping, piecewise linear methods, Taylor series expansion, and hybrid methods [2].

The selection of the approximation method and its hardware implementation are key aspects that constrain the accuracy of the activation function, and consequently, the learning and generalization capabilities of the whole ANN [3]. Moreover, too low accuracy leads to poor performance, while an excess of it unnecessarily increases hardware resources and reduces the processing speed. Despite the importance of proper specification of the accuracy of the activation function, very few works incorporate it as a design parameter [4]. To tackle this problem, a novel approximation scheme of the sigmoid function is proposed. The scheme is based on Taylor's Theorem and the Lagrange form of the remainder. A systematic design methodology which guarantees the accuracy of the approximation is provided.

Concerning digital hardware implementation of the approximation scheme, an original solution is proposed that reuses the circuitry of the linear part of the neuron. The circuit architecture is specially suited for FPGA-based implementations because it makes use of primitives and embedded resources that can be found in typical field programmable gate array (FPGA) families such as LUTs, and high performance low-power DSP (Digital Signal Processing) blocks. The main computation module of our solution is a single embedded DSP core. Firstly the core is dedicated to perform the computation of the linear part of the neuron, and then the same core is reused, with minor additional resources, to compute the sigmoid and obtain the output of the neuron.

Approximation Scheme: The approximation scheme divides the input range into two kinds of regions, the so called saturation regions and Taylor regions (see Fig. 1). The Taylor regions, in turn, are split into a number of intervals where a local approximation of the function is computed. The sigmoid function is given by

$$f(x)=\frac{1}{1+e^{-x}} \tag{1}$$

It can be approximated in any interval I containing a, $I=(a-r,a+r)$, with an n^{th} degree Taylor polynomial

$$f(x)\approx f(a)+f'(a)(x-a)+\frac{f''(a)}{2!}(x-a)^2+\cdots+\frac{f^{(n)}(a)}{n!}(x-a)^n \tag{2}$$

Where the approximation error (i.e. remainder) in I can be bounded using the Lagrange form of the remainder or error

$$|R_n(x)|\le\frac{|x-a|^{n+1}}{(n+1)!}M_n \text{, where } \left|f^{(n+1)}(x)\right|\le M_n \tag{3}$$

Equation (3) is very useful in practice because it provides a means of dealing with the maximum allowable approximation error ε as a design parameter. It is worth to mentioning that the output of the sigmoid function lies in the range (0,1), and verifies that $f(-x)=1-f(x)$, thus in the following only the positive semi-axis will be considered.

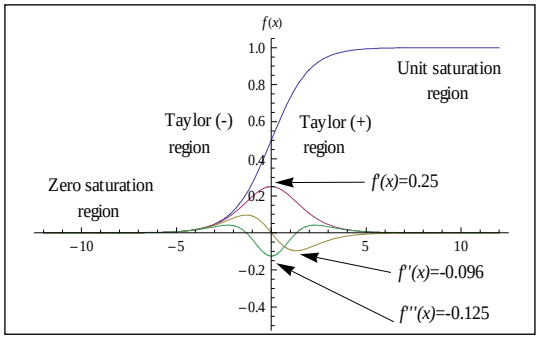

Fig. 1 *Sigmoid and extreme values of its first three derivatives.*

The saturation region is that region where the first derivative of the sigmoid function is close to zero (see Fig. 1). The starting point of the saturation region depends on the required precision. To determine where it starts, we have to find the value of x where the gap between '1' and the sigmoid function equals the maximum allowable error

$$1-f(x)=\frac{1}{1+e^{x}}=\varepsilon, \text{ with } 0.5\le f(x)<1 \tag{4}$$

Solving (4) results in $x=t=\ln(1/\varepsilon-1)$, being $t\ge 0$ the boundary between the Taylor region and the saturation region.

The positive Taylor region $[0,t]$ is split into ni intervals of width $2r$, then $r=t/2ni$. Replacing in (3) and taking into account that $|x-a|\le r \quad \forall x\in I$, the minimum number of intervals for a given error results

$$ni\ge\frac{1}{2}\left(\frac{M_n}{\varepsilon(n+1)!}\right)^{\frac{1}{n+1}}t \text{, with } t=\ln\left(\frac{1}{\varepsilon}-1\right) \text{ and } |R_n(x)|=\varepsilon \tag{5}$$

Table 1 provides the boundary of the Taylor region and the minimum number of intervals for different errors using a first order and a second order Taylor approximation scheme, according to (5). As can be seen, the larger the order of the polynomial the lower the required number of intervals in the Taylor region. In other words, the accuracy of the approximation scheme can be enhanced by using more terms in (2) or by refining the Taylor region segmentation.

Table 1: Design parameters (minimum values) as a function of the allowed error

Allowed error (ε)	Taylor region width (t)	Intervals 1th order series	Intervals 2nd order series	N_i	N_f	N_o
0.1	2.19	2	1	2	2	4
0.01	4.59	6	3	3	5	7
0.001	6.91	24	10	3	8	10
10^{-4}	9.21	102	28	4	12	14

Selection of the word-length: The proposed scheme has been implemented using a fixed-point fractional data format. The integer part of the input is represented by means of N_i bits while the fractional part requires N_f bits ($x_{N_i-1}\ldots x_0.x_{-1}\ldots x_{-N_f}$). The integer part depends on the width of the Taylor region: $2^{N_i}\ge t$, which can be written as

$$N_i\ge\frac{\ln t}{\ln 2}, N_i\in\mathbb{Z} \tag{6}$$

The fractional part of the input should be enough to represent changes in the inputs Δx that produce changes in the sigmoid function Δf equal to the maximum allowable error

$$\frac{\Delta f}{\Delta x} = \frac{\varepsilon}{2^{-N_f}}, \text{ for small increments } \frac{\Delta f}{\Delta x} \cong f'(x) \quad (7)$$

Taking into account that the first derivative reaches its maximum value at x=0, where $f'(0)$=0.25 (see Fig. 1)

$$2^{-N_f} \le \frac{\varepsilon}{0.25} = 4\varepsilon \text{ , then } N_f \ge -2 + \frac{|\ln \varepsilon|}{\ln 2}, \; N_f \in \mathbb{Z} \quad (8)$$

Concerning the output format, let N_o be the number of fractional bits used to represent the output $(0.y_{-1}\ldots y_{-N_o})$. Assuming that signal quantization is performed by truncation, the maximum quantization error is $\varepsilon_t = 2^{-N_o}$. Therefore

$$N_o \ge \frac{|\ln \varepsilon|}{\ln 2}, \; N_o \in \mathbb{Z} \quad (9)$$

Figure 2 depicts the approximation of the sigmoid function generated through the proposed scheme (5) to (9), for $\varepsilon = 0.01$ using a second order approximation scheme.

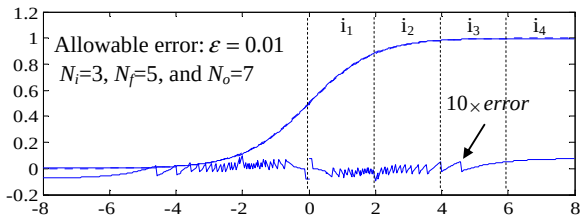

Fig. 2 *Reference sigmoid (—), second order approximation of the sigmoid (---), and error curve magnified by ten for* $\varepsilon = 0.01$.

Hardware implementation: The digital circuit depicted in Fig. 3 implements the sigmoid function using the second order approximation scheme. The input to the sigmoid circuit is the output of linear part of the neuron, but truncated to N_i+N_f bits. In this way, the sigmoid function outside the positive Taylor range is easily saturated to its maximum value. The main computation unit is a typical DSP core, configured to implement four different instructions [5]. Instructions 1 and 4 are firstly used to compute the linear stage of the neuron. Then, instructions 2 to 4 evaluate the sigmoid function.

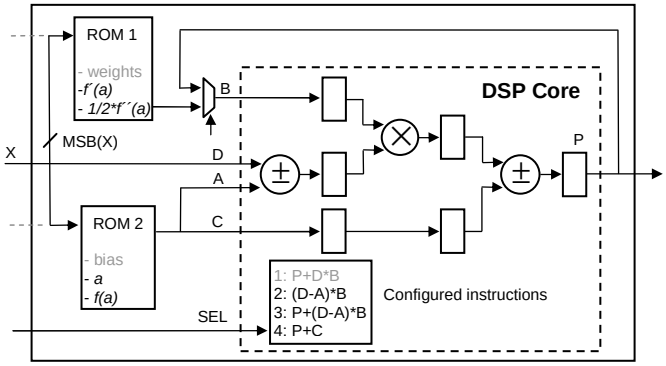

Fig. 3 *Sigmoid circuit (positive semi-axis). Resources in grey are used only in the computation of the linear part of the neuron.*

Two ROM memories are used, the same as in the linear stage of the neuron, ROM1 stores the sigmoid derivatives ($f'(a)$,$f''(a)$), while ROM2 is used to store the pairs of values (a,$f(a)$) -as many values as Taylor intervals. The memory words are addressed by means of the most significant bits of the input data. The second order scheme was implemented using a device of Xilinx's Virtex 6 family. The sigmoid circuit requires only 1 embedded DSP block, and 7 slices (11 LUTs), for a maximum allowable error $\varepsilon = 0.01$. Slice resources depend on the required word-length, which in turn depends on the allowable error. The circuit performs the computation of the sigmoid in only 7 clock cycles and is able to operate at 373.5 MHz.

Conclusion: A novel approximation scheme of the sigmoid function for efficient implementation of artificial neurons is proposed. The scheme provides a systematic way to guarantee the required accuracy as well as a very efficient solution for implementing the circuit using native resources of FPGAs. On the one hand, it reuses the circuitry of the linear stage of the neuron to compute the sigmoid function with minor additional resources. On the other hand, the use of embedded cores provides maximum processing speed with low power consumption.

Acknowledgments: This work was supported by the Spanish MINECO, and European FEDER funds (grant TEC2010-15388), and by the Basque Government (grants IT733-13, and S-PC12UN016).

I. del Campo, R. Finker, and J. Echanobe (*Department of Electricity and Electronics, Faculty of Sciences and Technology, University of the Basque Country, Leioa, Vizcaya, 48940 Spain*)
E-mail: ines@we.lc.ehu.es
K. Basterretxea (*Department of Electronic Technology, Technical Industrial Engineering School of Bilbao, University of the Basque Country, Bilbao, Vizcaya, 48013 Spain*)

References

1 Misra, J., and Saha, I.: 'Artificial neural networks in hardware: A survey of two decades of progress', *Neurocomputing*, 2010, **74**, pp. 239–255.
2 Armato, A., Fanucci, L., Scilingo, E.P., and De Rossi, D.: 'Low-error digital hardware implementation of artificial neuron activation functions and their derivative', *Microprocessors and Microsystems*, 2011, **35**, pp. 557–567.
3 Basterretxea, K., Tarela, J.M., del Campo, I., and Bosque, G.: 'An experimental study on nonlinear function computation for neural/fuzzy hardware design, *IEEE Trans. Neural Networks*, 2007, **18**, pp. 266–283.
4 Zamanlooy, B., and Mirhassani, M. 'Efficient VLSI implementation of neural networks with hyperbolic tangent activation function', *IEEE Trans. Very Large Scale Integration Systems*, accepted November 2012.
5. 'LogiCORE IP DSP48 Macro v2.1', DS754, http://www.xilinx.com/, accessed 12nd September 2013.