Neurocomputing 183 (2016) 39–55

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A massively parallel pipelined reconfigurable design for M-PLN based neural networks for efficient image classification

Nadia Nedjah [a,*], Felipe P. da Silva [a], Alan O. de Sá [a], Luiza M. Mourelle [b], Diana A. Bonilla [a]

[a] Department of Electronics Engineering and Telecommunication, Engineering Faculty, State University of Rio de Janeiro, Brazil
[b] Department of System Engineering and Computation, Engineering Faculty, State University of Rio de Janeiro, Brazil

ARTICLE INFO

Article history:
Received 30 August 2014
Received in revised form
22 May 2015
Accepted 25 May 2015
Available online 30 December 2015

Keywords:
Neural networks
Hardware
Multi-valued probabilistic node
Weightless neural network

ABSTRACT

Weightless Neural Networks (WNNs) are a powerful mechanism for pattern recognition. Aiming at enhancing their learning capabilities, Multi-valued Probabilistic Logic Nodes (M-PLN) are used, instead of crisp neurons with a 0/1 based RAM-nodes. An M-PLN stores a mapping of, or possibly, the triggering probability, for each input pattern that needs to be recognized. The M-PLN model attempts to strengthen the discrepancies between distinct patterns used during the training process and those that have not yet been processed. In this paper, an efficient yet customizable hardware architecture for M-PLN based neural network is proposed. It implements the learning and operation processes of a pyramidal network structure, augmented by a probabilistic rewarding/punishing search algorithm. The training algorithm can adapt itself to the overall hit ratio so far achieved by the network. Using class-dedicated layers, the hardware is able to handle image classification in parallel and thus, very efficiently. Furthermore, the classification process is performed in a pipelined manner so its stages never stop working until all input images are classified. Nonetheless, during network training, only one of these layers is activated. Last but not least, the architecture is customizable as its structure can be tailored in accordance to the application characteristics in terms of class number, pattern tuple size and image resolution. In order to evaluate the time and cost requirements of the proposed design, its underlying architecture was specified in VHDL and functionally tested. The presented results are two-fold: first, based on many functional simulations, estimated time and cost requirements are analyzed; second, to further assess the performance of the proposed the design, the VHDL model was synthesized to produce a semi-custom implementation on FPGAs. We also give an assessment of the quality of the entailed classification process. The architecture exhibits performance and reconfiguration capabilities that are very promising and encouraging towards the fabrication of a prototype on ASIC.

1. Introduction

Since artificial neural networks (ANNs) were proposed for the first time [16], the scientific, academic and industrial communities have been witnessing the advent of many derivative models. When the absence of a consistent mathematical model imposes limits for computationally viable solutions, ANNs stand for a well-succeeded alternative in solving problems wherein learning abilities, such as cognition and perception, are necessary. For the resolution of such class of problems, the human brain structure seems more appropriate, since it works in a parallel manner, besides the capacity to learn by examples.

The computational model behind artificial neurons shares the same concepts of parallelism and non-algorithmic learning capacity of their biological counterpart in human brains. This capacity allows for the data generalization and association in ANNs. Based on a given training set of patterns, a neural network can recognize new patterns with similar characteristics, even though these have not been seen during the training process. This leads to a high degree of fault tolerance, when input data present some noise. Several neural network models were proposed [16,3,8]. In most of these models, the network inputs and partial results are multiplied by some associated weights. This makes hardware implementation of ANNs inefficient and thus keeps away the use of ANNs in industrial devices. Consequently, Weightless Neural Networks (WNNs) appeared to fill in this gap [3]. WNNs are memory based, which makes them suitable for hardware implementation. Embeddable and customizable hardware designs for computational intelligence oriented techniques, such as artificial neural networks [12,23,22,30], fuzzy logic [21], genetic algorithms [24] and particle swarm optimization [10], among others, are very attractive to use aiming at reducing time-to-market availability of devices that require such a support. Recent research works take

* Corresponding author.

http://dx.doi.org/10.1016/j.neucom.2015.05.138
0925-2312/

advantage of WNNs in a variety of applications, such as image recognition [5,9], brain waves identification [2], financial time series analysis [15], and even in machine consciousness [28].

This paper proposes a massively parallel and pipelined novel design for Multi-valued Probabilistic Logic Node based neural networks (M-PLN NNs). An efficient and customizable architecture is presented. It implements the structure and learning process of a pyramidal NN, augmented by a probabilistic rewarding/punishing search algorithm. The training algorithm can adapt itself to the overall hit ratio achieved by the learning process. During test and operation, the design is massively parallel and explores a pipeline to further accelerate image classification. The proposed architecture was specified in VHDL and functionally tested. A VHDL parametrizable code for the architecture, in terms of image and class number, pattern tuple size and image resolution, is used to evaluate the impact of each of these parameters on the design performance. It is noteworthy, at this stage, to point out that the contribution of this paper resides mainly in the proposed architecture. As far as we know, there is no existing hardware architecture for M-PLN NNs. The code specification of the hardware model, which describes the structure and behavior of the design, is laborious but fairly straightforward in VHDL or any other HDL, such as Verilog and HardwareC. In this case, the HDL choice is as almost always, oriented by the availability of the required development kits. This said, however, the detailed architectural description presented allows any interested reader to reproduce the design in VHDL or in any other preferred HDL. Yet, the VHDL code of the design will be made available upon a contact with one of the authors.

The performance characteristics of the proposed design are thoroughly presented and analyzed. Firstly, we estimate the time and cost requirements based on many functional simulations of the VHDL model. The impacts of the number of images used, the number of classes intended, the size of pattern tuple and the image resolutions on time and cost requirements are presented and discussed. Furthermore, note that the image and class numbers used during training and operation are co-related. This is also the case for the image resolution and tuple size. Hence, we also evaluate and analyze the impact of pairs of image and class numbers as well as pairs of image resolutions and tuple sizes on the design performance and cost. Secondly, to further assess the performance of the proposed the design, the VHDL model was synthesized to produce a semi-custom implementation on FPGAs. It is shown that the architecture exhibits performance, cost requirements and reconfiguration capabilities that are very promising as well as encouraging towards the fabrication of a prototype on ASIC.

The rest of this paper is organized as follows: First, in Section 2, we introduce some basic concepts of weightless neural networks, including the model generalization to PLN and MPLN based networks. Thereafter, in Section 3, we present the macro-architecture of the proposed design. Subsequently, in Section 4, we give details about the micro-architecture. There follows, in Section 5, a thorough explanation of the operational modes of the design. After that, in Section 6, we provide a comprehensive analysis and discussion of time and cost requirements entailed by the proposed design based on simulation results augmented by an evaluation of a semi-custom implementation of the design on FPGAs. Moreover, we provide an assessment of the classification quality of the obtained prototypes. In Section 7, we conclude the paper and point out some directions to complete and improve the work done so far. Note that a preliminary version of this work has appeared in [29].

2. Weightless neural networks

The concept of weightless neural networks started with the so-called n-tuple classification machine [9], also known as RAM-based neural networks. These do not associate learned knowledge with the connections that link the neurons, as it is the case in ANNs. Instead, the knowledge is stored in simple random access memories available internally at each neuron. The neuron binary inputs are used to compose an address, which is used to access a specific location in the neuron's RAM. The content of the addressed RAM location is used as the neuron's result. In the case of a RAM-based neural network, the training can be done in a single step, basically storing the expected output at the RAM location whose address is the binary input combination. The RAM-based neural networks gained interest over time and evolved to produce the so-called weightless neural networks [8]. Note that the concept of such neural network architectures as weightless has been first introduced in [6], but identified as such later on. So, the main difference between WNNs and other models of ANNs consists of how the learned information is tracked in. In contrast with conventional ANNs, wherein the learned information is synthesized via weight values that are associated with the connections between the neurons of different layers, in WNNs a similar information is simply stored in random access memories available inside the neurons. In order to improve the learning capabilities of such WNNs, more and more sophisticated information, ranging from a single 0/1 bit to a floating-point values, are stored in the neuron internal memory. Two of these improvements, identified as Probabilistic Logic Node (PLN) [13] and Multi-valued Probabilistic Logic Node (M-PLN) [25] based models, are described in the following sections. Recall that, the herein proposed architecture is tailored for M-PLN, which is the most general and effective in terms of learning capabilities.

2.1. Probabilistic logic node based neural networks

A PLN is a RAM-based neuron that includes a local memory whose entries may contain three possible combinations: one combination to represent 0, one to represent 1 and a third combination to represent the so far unseen or unknown pattern [13], which is termed u. It follows that, instead of 1-bit memory cells, the PLN internal memory need to be formed of 2-bit cells. Furthermore, all the nodes of a PLN-based network are initialized using the 2-bit combination associated with u, indicating that so far all patterns are unknown to the node. The key idea behind the PLN model consists of strengthening the difference between the patterns that have been seen during the training phase and those not yet seen. Therefore, the output function, instead of deterministic, like in the simple WNN model, such as that used in WiSARD [4,5], becomes probabilistic. In this case, the neuron output for pattern p, denoted by $\mathcal{O}_p$, can be defined according to:

$$\mathcal{O}_p = \begin{cases} 0 & \text{if } M[p] = \mathcal{C}(0) \\ 1 & \text{if } M[p] = \mathcal{C}(1) \\ \varrho(0,1) & \text{if } M[p] = \mathcal{C}(u), \end{cases} \tag{1}$$

wherein $M[p]$ is the content of the memory cell associated to pattern p in the node memory M, $\mathcal{C}(v)$ is a function that maps the probabilistic values $v \in \{0, 1, u\}$ to a distinct binary combination and $\varrho(0,1)$ is a function that generates either 0 or 1 according to a uniform distribution, yielding 0s and 1s with the same probability.

The use of the concept of unknown value u doubles the memory size required in a simple WNNs, such as in WiSARD. This is because the stored information must be of 2 bits. As there are 4 possible combinations available, one can choose indifferently to use either combination $\mathcal{C}(u) = 10$ or $\mathcal{C}(u) = 11$ to represent the logic value of u, with combinations $\mathcal{C}(0) = 00$ and $\mathcal{C}(1) = 01$ representing 0 and 1, respectively. Note its is not mandatory to have