A Systematic Literature Review of Hardware Neural Networks

Dorfell Parra
School of Electrical and
Electronic Engineering
Universidad Nacional de Colombia
Bogotá, Colombia
Email: dlparrap@unal.edu.co

Carlos Camargo
School of Electrical and
Electronic Engineering
Universidad Nacional de Colombia
Bogotá, Colombia
Email: cicamargoba@unal.edu.co

***Abstract*—Although Neural Networks (NN) are extremely useful for the solution of several problems such as object recognition and semantic segmentation, the NN libraries usually target devices which face several drawbacks such as memory bottlenecks and limited efficiency (e.g. GPUs, multi-core processors). Fortunately, the recent implementation of Hardware Neural Networks aims to tackle down this problem and for that reason several researchers had turn back their attention to them. This paper presents the Systematic Literature Review (SLR) of the most relevant HNN works presented in the last few years. The main sources chosen for the SLR were the IEEE Computer Society Digital Library and the SCOPUS indexing system, from which 61 papers were reviewed according to the inclusion and exclusion criteria, and of which after a detail assessment, only 20 papers remained. Moreover, the increase in the number of papers per year reflects that the interest in HNN had been growing up. Finally, the results show that the most popular NN hardware platforms are the FPGAs-based.**

Keywords - HNN, SLR, Framework, FPGA, Neural Networks.

I. Introduction

With the appearance of high performance platforms in the last decade (i.e. GPU, FPGAs), Neural Networks (NNs) are becoming an attractive tool for many classification and object recognition problems. Despite this, the existent APIs for running NNs don't exploit the hardware resources efficiently and for this reason several researches have turn back their attention to Hardware Neural Networks (HNNs) [20], [23], [26]. HNNs are the implementation of accelerators architectures that evaluate the NNs forward propagation algorithms, by using reconfigurable platforms like FPGAs, or hardcore designs like the VLSI circuits [9], [27]. Usually, the networks are trained by means of tools such as Caffe [1], MATLAB [2], TensorFlow [3], Microsoft Cognitive Toolkit [4], etc. and then, with the weights and bias calculated, a hardware accelerator is implemented [18], [21], [24]. Nevertheless, the discussion of how to design HNNs had been widen due to number of works that had been proposed in the last few years [9], and up to day, a new literature review is needed to identify the most relevant search streams that had appeared lately, the NN types being used, the organizations leading the research, the research limitations and the quality of the implementation frameworks being proposed. In this paper, we aim to answer all of these questions by means of a Systematic Literature Review (SLR) [7], [8]. This paper is organized as follows; Section II gives a brief background information on HNN and implementation frameworks. Section III describes the carried out Systematic Literature Review. Section IV presents the SLR results and Section V the discussion. Finally, the conclusion is drawn in Section VI.

II. Background

A. Hardware Neural Networks (HNNs)

Currently, neural networks are implemented by means of software libraries such as TensorFlow [3] and Microsoft Cognitive Toolkit [4], but in the last few years there had been a growing interest in hardware implementations of neural networks, which are known as Hardware Neural Networks (HNNs) [9]. HNN aim to make the forward evaluation process in a hardware platform, by using the weights and biases previously computed with the software libraries in the training process. HNN topologies also receive the name of accelerators, and they are mainly composed by Processing Elements (PEs), memory units and kernel processing units [6], [12], [14], [15], [19],[27],[31], as it is shown in Figure 1.

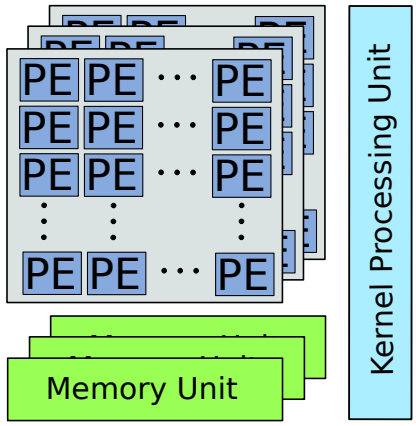

Fig. 1: Neural Network Accelerator Circuit. Adapted from [15], [27], [31].

978-1-5386-6740-8/18/$31.00

Hence, the PEs are hardware cores in charge of computing the neuron basic operations (i.e. floating or fixed point multiplication and addition), the memory units are used to store each neuron output or the activation function when used as a look-up table [12], [15], [19], and the kernel processing unit manages all the aforementioned resources to compute the NN forward propagation [31].

B. HNN Implementation Framework

Implementing neural networks in hardware is a task that involve many stages, and that can end up becoming very complex to manage. For these reasons, several attempts to propose frameworks that aim to integrate all of those stages and reduce the inner complexity had appeared in the last few years [11], [23],[26]. Usually, a framework processing flow includes some basic stages like training the neural network, analyzing the NN architecture and generating the synthesizable HDL files. A brief description of those basic stages is given below.

- **Stage 1. Creating the NN:** The NN parameters are defined (e.g.NN type, the number of layers and neurons, and the activation function, etc.) and passed to a software library like TensorFlow [3] or Microsoft Cognitive Toolkit [4].
- **Stage 2. Design Space Exploration:** By using the hardware platform specifications, the weights and biases computed by the library, and the aforementioned parameters, the framework will make a design space exploration (DSE) [23] to determine the amount of resources needed, and the hardware constraints imposed by the platform.
- **Stage 3. HDL files Generation:** In this stage the HNN design is mapped into the hardware platform, and the accelerator HDL source files will be generated.
- **Stage 4. Testing the HNN:** In the last stage the configuration files are created, and the HNN performance can be tested on the platform.

III. Research Methodology

The literature review was made by applying the Systematic Literature Review (SLR) method, proposed by Kitchenham et al. in [7] and [8]. The goal of this review is to identify the gaps existing in the HNN implementation process by studying the relevant works in the state of the art. Additionally, according to the literature NN implementations on traditional platforms like general purpose processors don't use hardware resources efficiently, and therefore there is a growing interest in exploring others platforms [20], [23], [26]. On the other hand, Graphical Processing Units (GPUs) are being widely used for training NN because of their high throughput, but those implementations are limited by the local memory available and the communication bandwidth between the host and the GPU, [16], [17], [26] besides, its power consumption make them not practicals for embedded systems. Fortunately, these implementations can be improved by using accelerators in custom hardware (e.g. ASICs, FPGA), which have demonstrated to have better performance during the feedforward prediction stage and low power consumption [26], [30]. Moreover, FPGAs are easily programmable and inexpensive, these being why they are the most advantageous option. Furthermore, with the appearance of more optimization design techniques and the amount of FPGA logic resources available being increased, the design exploration space is enlarged boosting the HNN implementation on FPGA-based platforms. By taking into account the above-mentioned factors, it was decided to focus this literature review on HNN works and implementation frameworks designed for these platforms, without including GPU-based works. Lastly, the SLR method steps are documented below.

A. Research Questions (RQ)

The research questions addressed by this review are:

- RQ1: How many frameworks for implementing HNN has been proposed since 2010?
- RQ2: What types of Neural Networks are being addressed?
- RQ3: What individuals and organizations are leading the research?
- RQ4: What are the limitations of current research?
- RQ5: Is the quality of the implementation frameworks improving?

With respect to RQ1, the review starts at 2010 because HNNs start to be a feasible option when the resources and computation platforms available were sufficient for the efficient implementation around 2010. With respect to RQ2, there are several types of NN that can be implemented in hardware, however, differences in input data, classification tasks, activation function, etc. can lead to choose one instead of another. For this reason it is important to know which are the NN types being used for HNN. With respect to RQ3, it is essential to identify HNN research trends, relevant authors and leading works to be aware of the current problems being studied. With respect to limitations of HNN research (RQ4) the following issues are going to be considered:

- RQ4.1: Were the scope of HNN implementation frameworks limited?
- RQ4.2: Is there evidence that the use of HNNs is limited due to lack of implementation frameworks?
- RQ4.3: Is the quality of implementation frameworks appropriate?
- RQ4.4: Are frameworks contributing to the implementation of HNN by defining practice guidelines?

With respect to RQ5, it is important to know if the proposed frameworks are being improved in subsequent works or if the new proposed frameworks are taking different approaches.

B. Search Process

The IEEE Computer Society Digital Library, the SCOPUS indexing system and Open Access organizations like IAES, IOSR were used in the search process. All searches were based on titles, keywords and abstracts of works published in journals, conferences and symposiums since 2010, which

are shown in Table I. The search string used in the IEEE library was "Neural Networks" AND "Hardware" and the search string used in SCOPUS was TITLE-ABS-KEY("Neural Networks") AND TITLE-ABS-KEY("Hardware") OR TITLE-ABS-KEY("Framework").

C. Study selection

The results for the different searches were added, obtaining a total number of 491 papers published between Jan 1st 2010 and March 31th 2017: 143 from the IEEE digital library, 343 from SCOPUS indexing system and 5 from the open access organizations. To these papers, the following inclusion and exclusion criteria were applied.

Topics used to include the papers:

- Frameworks for implementing HNN with defined research questions, search process, data extraction and data presentation, whether or not the researchers referred to their study as a implementation framework.
- Approach to optimize current implementations of HNN.

Papers on the following topics were excluded:

- Informal literature surveys (no defined research questions; no defined search process; no defined data extraction process).
- Papers presenting implementations of HNN with not defined procedures or not discussing the procedures used.
- Papers presenting GPU-based works, which are limited by local memory constraints and bandwidth communication will also be excluded.
- Duplicate reports of the same study (when several reports of a study exist in different journals the most complete version of the study was included in the review).

After excluding papers that were obviously irrelevant, had not enough information, or were duplicates, there were 61 papers remaining. Those papers were then subject to a more detailed assessment, where each paper was reviewed to identify papers that could be rejected on the basis that they did not include literature reviews, or that they were not related to implementation frameworks. This led to the exclusion of 41 papers. The remaining papers are shown in Table II.

D. Quality assessment (QA)

Each article was evaluated using the following quality assessment (QA) questions based on [7]:

- QA1: Is the HNN implementation process presented explicitly?
- QA2: Is the literature search likely to have covered all relevant studies?
- QA3: Did the HNN implementation assess the quality/-validity of previous included studies?
- QA4: Were the basic data/studies adequately described?

There are three possible outputs for each question with the following score: $Y\ (yes) = 1$, $P\ (partly) = 0.5$ and $N\ (no) = 0$. For QA1: Y the implementation process is presented explicitly, P the implementation process is implicit and N the implementation process is not defined and cannot be readily inferred.

For QA2: Y the authors had cited at least 20 works including highly cited works, P the authors had cited between 15 and 19 works including relevant works. N the authors had cited less than 15 works or they had cited irrelevant works.

For QA3: Y The HNN implementation performance improved former ones in more than 2x, P the performance was less than 2x of previous ones, and N performance was not reported.

For QA4: Y information of each primary study is presented, P each primary study is barely presented, and N any information of each primary study is given.

E. Data collection

The data extracted from each work were:

- The source (journal, conference or symposium) and full reference.
- Classification of the study type (i.e. HNN implementation, VLSI design, HNN implementation frameworks).
- Main topic area.
- The author(s), their institution and the country where it is situated.
- Summary of the study including the main research questions and the answers.
- Research question/issue.
- Quality evaluation.
- How many primary studies were used in the work.

F. Data analysis

The data was tabulated (see Tables I and III) to show:

- The number of HNN works published per year and their source (addressing RQ1).
- Whether the HNN work referenced others papers (addressing RQ1).
- The topics studied by the HNN works, i.e. HNN implementation, VLSI design, HNN implementation frameworks (addressing RQ2 and RQ4.1).
- The authors: the affiliations of the authors and their institutions was reviewed but not tabulated (addressing RQ3).
- The number of previous HNN works in each paper (addressing RQ4.2).
- The quality score for each HNN work (addressing RQ4.3).

IV. Results

A. Search Results

Including works from journals, symposiums and conferences they were 20 papers reviewed. These papers are shown in Table II.

B. Quality evaluation of the HNN works.

The HNN works shown in Table II were assessed based on the quality assessment (QA) questions. These results are shown in Table III.

Source	Acronym	Organization	Publication
Transactions on Neural Networks and Learning Systems	TNNLS	IEEE	Journal
Transactions on Very Large Scale Integration Systems	TVLSIS	IEEE	Journal
Transactions on Computers	TC	IEEE	Journal
Neural Networks	NN	ELSEVIER	Journal
Neurocomputing	NC	ELSEVIER	Journal
Information Fusion	IF	ELSEVIER	Journal
Computer Methods and Programs in Biomedicine	CMPB	ELSEVIER	Journal
Pattern Recognition	PR	ELSEVIER	Journal
Engineering Applications of Artificial Intelligence	EAAI	ELSEVIER	Journal
Engineering Research And Development	IJERD	Peer Reviewed	Journal
Electrical and Computer Engineering	IJECE	IAES	Journal
Electronics and Communication Engineering	JECE	IOSR	Journal
International Conference on Data Mining Workshops	ICDMW	IEEE	Conference
International Conference on Computer Science and Network Technology	ICCSNT	IEEE	Conference
International Conference on Architectural Support for Programming Languages and Operating Systems	ASPLOS	IEEE/ACM	Conference
International Conference on Parallel Architectures and Compilation Techniques	PACT	IEEE/ACM	Conference
Annual International Symposium on Computer Architecture	ISCA	IEEE/ACM	Symposium
Annual International Symposium on Field Programmable Custom Computing Machines	ISFPCCM	IEEE	Symposium
Annual International Symposium on Field Programmable Gate Arrays	FPGA	IEEE/ACM	Symposium

TABLE I: Selected journals, symposiums and conference proceedings.

Study Ref.	Authors	Date	Paper type	Number primary studies	Review topics
[9]	Misra & Saha	2010	Journal	278	Overview of HNN models: HNN chips, Cellular HNN, Neuromorphic HNN, Optical NN.
[11]	Farabet et al.	2011	Conference	27	HNN implementation, hardware architectures.
[12]	Rana D. Abdu-Aljabar	2012	Journal	23	HNN implementation.
[14]	Shakoory	2013	Journal	12	HNN implementation.
[15]	Mohammed et al.	2013	Journal	9	HNN implementation.
[18]	Chen et al.	2014	Conference	44	HNN implementation, VLSI design.
[19]	Singh et al.	2015	Journal	11	HNN implementation.
[20]	Zhang et al.	2015	Symposium	16	HNN implementation.
[21]	Zhou, Y., & Jiang, J.	2015	Conference	12	HNN implementation.
[22]	Du et al.	2015	Symposium	61	HNN implementation, VLSI design.
[23]	Venieris et al.	2016	Symposium	16	HNN implementation framework.
[24]	Murakami, Y.	2016	Conference	8	HNN implementation.
[25]	Motamedi et al.	2016	Conference	10	HNN Parallelism.
[26]	Dundar et al.	2016	Journal	49	HNN implementation.
[27]	Li et al.	2016	Symposium	11	HNN implementation.
[28]	Saldanha et al.	2016	Symposium	13	HNN implementation.
[29]	Wang et al.	2016	Symposium	19	HNN implementation, VLSI design.
[30]	Ortega-Zamorano et al.	2016	Journal	44	HNN implementation framework for Backpropagation.
[31]	Kyrkou et al.	2016	Journal	41	HNN implementation
[32]	Luo et al.	2017	Journal	66	HNN implementation, VLSI design.

TABLE II: Systematic Review of HNN Studies.

C. Quality factors

The average Quality Scores (QS) for studies each year, the mean and the standard deviation σ are shown in Table IV. As can be seen the number of HNN studies in the last few years has grew up from 1 study per year up to 9 studies, showing the growing interest for HNN. Also, the average QS per year

Study Ref.	Paper type	QA1	QA2	QA3	QA4	Total Score
[9]	Journal	N	Y	N	Y	2.0
[11]	Conference	Y	Y	Y	P	3.5
[12]	Journal	Y	P	P	Y	3.0
[14]	Journal	Y	N	P	P	2.0
[15]	Journal	N	N	N	N	0.0
[18]	Conference	Y	Y	Y	Y	4.0
[19]	Journal	Y	N	P	N	1.5
[20]	Symposium	Y	P	Y	P	3.0
[21]	Conference	P	P	P	P	2.0
[22]	Symposium	Y	Y	Y	Y	4.0
[23]	Symposium	Y	P	Y	Y	3.5
[24]	Conference	Y	N	P	P	2.0
[25]	Conference	Y	P	Y	Y	3.5
[26]	Journal	Y	Y	Y	Y	4.0
[27]	Symposium	Y	P	P	N	2.0
[28]	Symposium	P	P	P	N	1.5
[29]	Symposium	P	P	P	P	2.0
[30]	Journal	Y	Y	Y	Y	4.0
[31]	Journal	Y	Y	Y	Y	4.0
[32]	Journal	Y	Y	Y	Y	4.0

TABLE III: Quality evaluation of the HNN studies.

has been quasi-stable around 3.0 (i.e. 2.88), which can be seen as an increase in the number of most comprehensive works on the topic.

	Year							
	2010	2011	2012	2013	2014	2015	2016	2017
# of papers	1	1	1	2	1	4	9	1
QS Mean	2.0	3.5	3.0	1.0	4.0	2.625	2.94	4
QS σ	0.88	0.625	0.12	1.88	1.12	0.255	0.06	1.12

TABLE IV: Average Quality Scores (QS) for studies by publication date.

V. Discussion

The answers to the research questions are discussed in this section.

- RQ1: How many frameworks for implementing HNN has been proposed since 2010?
  The revision of several studies from 2010 to 2017 lead to 20 relevant HNN studies. Moreover, it is observed that the interest in HNN is growing up as the number of studies per year.
  Relevant studies included 1 survey of the HNN implementations proposed before 2010 [9], 4 studies of HNN Very Large Scale Integration (VLSI) designs [18], [22], [32], [29], and the rest of studies proposed an HNN design implemented in a reconfigurable platform (e.g. FPGA). Despite not all of them presented an explicit framework, the implementation process could be readily inferred.
- RQ2: What types of Neural Networks are being addressed?
  Most of the works aimed to implement Convolutional Neural Networks (CNN) [5], [10]. CNN has been highly accepted in classification and object recognition problems because of its accuracy and relatively fair cost. CNNs are a class of Deep Neural Networks (DNN) where weights are shared across neurons, thus reducing the memory needed to stored the training parameters.
- RQ3: What individuals and organizations are leading the research?
  The leadership of HNNs implementation can be divided by approach. The VLSI design of HNNs is lead by the work group form by the State Key Laboratory of Computer Architecture in China, the Institute of Computing Technology processing in China and the Inria Institution in France. They had designed and implemented at least 4 different HNN chips [18], [22], [29].
  On the other hand, there are different authors that had contributed with several studies about HNNs implementation in reconfigurable platforms. For example, Eugenio Culurciello from the Courant Institute of Mathematical Sciences, New York University and Yann LeCun from the Electrical Engineering Department, Yale University both in USA presented works that include a dataflow processor for vision [11], and an Embedded Streaming DNN Accelerator [26]. In addition, Stylianos Veneris and Christos-Savvas Bougaris from the Department of Electrical and Electronic Engineering, Imperial College in London had presented fpgaConvNet, a framework for mapping CNN on FPGAs in [23]. Finally, Francisco Ortega-Zambrano et al. from the Departamento de Lenguajes y Ciencias de la Computación, Universidad de Málaga in Spain had proposed an efficient implementation of the NN training stage on FPGA in [30].
- RQ4: What are the limitations of current research?
  Currently, due to the number of computational resources (i.e. memory and processing) needed by HNNs implementations, researches aim to commercials FPGA-based systems (i.e. mostly large FPGAs), while the design of HNNs VLSI chips remains limited. Moreover, there are others factors that restrict the VLSI research such as: the power consumption, die size, fabrication technology and cost. Another important reason is that there is not a general agreement between approaches and there are studies that consider parameters that are not important to others studies, widening the exploration space but reducing the concentrated efforts. Thus, there are a few implementation frameworks proposed and their quality variates from barely acceptable to regular.
- RQ5: Is the quality of the implementation frameworks improving?
  Due to the growing interest in implementing HNN, the number of proposed frameworks by year has been increasing, and so the quality of studies published. For example, current studies offer a more complete description of the proposed work, make comparisons with similar studies and provide external links that widen the information available.

VI. Conclusions

This article presented a Systematic Literature Review of the latest works related to the implementation of neural networks

into hardware, from which two main streams can be identified: HNNs VLSI designs and HNNs implementation in reconfigurable platforms. Hence, the last one has the majority of studies due to the known constrictions of the VLSI fabrication process. Also, it was found that CNNs are the most aimed NN because of its features and applications. In addition, the awaken interest in HNNs has revealed the necessity of implementation frameworks, that allow to identify relevant parameters and that present a solid number of stages for accomplish the implementation. Unfortunately, frameworks available in the state of the art lack of simplicity, usually aim to bigger hardware platforms, have an excessive use of logic resources and present an acceptable accuracy.

Moreover, there are several problems in the implementation process that still had to be tackle down like: memory bottlenecks, scarce number of resources, complexity of the NNs, implementation precision and accuracy, and efficient HNNs training.

REFERENCES

[1] *Caffe: Deep Learning Framework.* [Online]. Available: http://caffe.berkeleyvision.org/, accessed Feb. 15, 2018.

[2] *MathWorks: MATLAB.* [Online]. Available: https://www.mathworks.com/products/matlab.html, accessed Feb. 15, 2018.

[3] *TensorFlow: An open-source software library for Machine Intelligence.* [Online]. Available: https://www.tensorflow.org/, accessed Feb. 15, 2018.

[4] *Microsoft Cognitive Toolkit.* [Online]. Available: https://www.microsoft.com/en-us/cognitive-toolkit/, accessed Feb. 15, 2018.

[5] *Face Image Analysis With Convolutional Neural Networks.* [Online]. Available: https://lmb.informatik.uni-freiburg.de/papers/download/du_diss.pdf, accessed Feb. 15, 2018.

[6] A. Muthuramalingam, S. Himavathi & E. Srinivasan, "Neural Network Implementation Using FPGA: Issues and Application", in Journal of Information Technology, 4(2), 86-92, 2008.

[7] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, & S. Linkman, "Systematic literature reviews in software engineering - A systematic literature review", in Information and Software Technology, 51(1), 7-15, Nov. 2008.

[8] B. Kitchenham,R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi, & S. Linkman, "Systematic literature reviews in software engineering-A tertiary study", in Information and Software Technology, Aug. 2010.

[9] J. Misra, & I. Saha,"Artificial neural networks in hardware: A survey of two decades of progress", Neurocomputing, vol. 74, Issues 13, pages 239-255, Dec. 2010.

[10] Z. Saidane, "Image and video text recognition using convolutional neural networks: Study of new CNNs architectures for binarization, segmentation and recognition of text images" in LAP LAMBERT Academic Publishing 2011.

[11] C. Farabet, B. Martini, B. Corda, P. Akselrod, E. Culurciello, & Y. Lecun, "NeuFlow: A runtime reconfigurable dataflow processor for vision", in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 109-116, Jun. 2011.

[12] Mrs. Rana and D. Abdu-Aljabar, "Design and Implementation of Neural Network in FPGA", in Journal of Engineering and Development, vol. 16, No.3, Sep. 2012.

[13] *Survey: Implementing Dense Neural Networks in Hardware.* [Online]. Available: https://pdfs.semanticscholar.org/b709/459d8b52783f58f1c118619ec42f3b10e952.pdf, accessed Feb. 15, 2018.

[14] G. H. Shakoory, "FPGA Implementation Of Multilayer Perceptron For Speech Recognition", in Journal of Engineering and Development, vol. 17, No.6, Dec. 2013.

[15] E. Z. Mohammed and H. K. Ali, "Hardware Implementation of Artificial Neural Network Using Field Programmable Gate Array", in International Journal of Computer Theory and Engineering, vol. 5, No. 5, Oct. 2013.

[16] A. krizhevsky, *One weird trick for parallelizing convolutional neural networks.* [Online]. Available: https://arxiv.org/abs/1404.5997, accessed Feb. 15, 2018.

[17] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, & E. Shelhamer E, *cuDNN: Efficient Primitives for Deep Learning.* [Online]. Available: https://arxiv.org/abs/1410.0759, accessed Feb. 15, 2018.

[18] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, & O. Temam, "DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning", in Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS'14, 269-284, Mar. 2014.

[19] S, Singh, S. Sanjeevi, S. V, A. Talashi,"FPGA Implementation of a Trained Neural Network", in IOSR Journal of Electronics and Communication Engineering (IOSR-JECE). vol. 10, Issue 3, ver. III May/Jun. 2015.

[20] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, & J. Cong, "Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks", in Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays - FPGA'15, 161-170, Feb. 2015.

[21] Y. Zhou, & J. Jiang, "An FPGA-based accelerator implementation for deep convolutional neural networks", in 4th International Conference on Computer Science and Network Technology (ICCSNT), Harbin, pp. 829-832, Dec. 2015.

[22] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, O. Temam, "ShiDianNao: Shifting Vision Processing Closer to the Sensor", in Proceedings of the 42nd Annual International Symposium on Computer Architecture-ISCA'15, 92-104, Jun. 2015.

[23] S. I. Venieris, & C. S. Bouganis, "FpgaConvNet: A Framework for Mapping Convolutional Neural Networks on FPGAs", in Proceedings - 24th IEEE International Symposium on Field-Programmable Custom Computing Machines, FCCM 2016 (pp. 40-47), May. 2016.

[24] Y. Murakami, FPGA implementation of a SIMD-based array processor with torus interconnect. In 2015 International Conference on Field Programmable Technology, FPT 2015 (pp. 244-247), (2016. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/FPT.2015.7393159

[25] Motamedi, M., Gysel, P., Akella, V., & Ghiasi, S. (2016). Design Space Exploration of FPGA-Based Deep Convolutional Neural Networks. In 21st Asia and South Pacific Design Automation Conference (pp. 575-580). https://doi.org/10.1109/ASPDAC.2016.7428073

[26] Dundar, A., Jin, J., Martini, B., & Culurciello, E. (2016). Embedded Streaming Deep Neural Networks Accelerator With Applications. in IEEE Transactions on Neural Networks and Learning Systems, vol.PP, no.99, (pp.1-12). https://doi.org/10.1109/TNNLS.2016.2545298

[27] Li, N., Takaki, S., Tomioka, Y., & Kitazawa, H. (2016). A Multistage Dataflow Implementation of a Deep Convolutional Neural Network Based on FPGA For High-Speed Object Recognition. In 2016 IEEE Southwest Symposium On Image Analysis and Interpretation (SSIAI) (pp. 165-168). 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE. https://doi.org/10.1109/SSIAI.2016.7459201

[28] Saldanha, L. B., & Bobda, C. (2016). "Sparsely connected neural networks in FPGA for handwritten digit recognition", in Proceedings - International Symposium on Quality Electronic Design, ISQED, pp. 113-117, May. 2016.

[29] Y. Wang, L. Xia, T. Tang, B. Li, S. Yao, M. Cheng, & H. Yang, "Low Power Convolutional Neural Networks on a Chip", in IEEE International Symposium on Computer Architecture, (1), 129-132, Apr. 2016.

[30] F. Ortega-Zamorano, J. M. Jerez,D. U. Munoz, R. M. Luque-Baena, & L. Franco, "Efficient Implementation of the Backpropagation Algorithm in FPGAs and Microcontrollers", in IEEE Transactions on Neural Networks and Learning Systems, 27(9), 1840-1850, Aug. 2016.

[31] C. Kyrkou, C. S. Bouganis, T. Theocharides, M. M. Polycarpou,"Embedded Hardware-Efficient Real-Time Classification With Cascade Support Vector Machines", in IEEE Transactions on Neural Networks and Learning Systems. vol. 27, no. 1, Jan. 2016.

[32] T. Luo, S. Liu, L. Li, Y. Wang, S. Zhang, T. Chen, Z. Xu, O. Temam, Y. Chen, "DaDianNao: A Neural Network Supercomputer", in IEEE Transactions on Computers, vol. 66, no. 1, pp. 73-88, Jan. 1 2017.