electronics

MDPI

Article

Energy Efficiency of Machine Learning in Embedded Systems Using Neuromorphic Hardware

Minseon Kang, Yongseok Lee and Moonju Park *

Department of Computer Science & Engineering, Incheon National University, Incheon 22012, Korea; kangpenguin@naver.com (M.K.); dyd9422@gmail.com (Y.L.)

* Correspondence: mpark@inu.ac.kr

Received: 25 May 2020; Accepted: 29 June 2020; Published: 30 June 2020

Abstract: Recently, the application of machine learning on embedded systems has drawn interest in both the research community and industry because embedded systems located at the edge can produce a faster response and reduce network load. However, software implementation of neural networks on Central Processing Units (CPUs) is considered infeasible in embedded systems due to limited power supply. To accelerate AI processing, the many-core Graphics Processing Unit (GPU) has been a preferred device to the CPU. However, its energy efficiency is not still considered to be good enough for embedded systems. Among other approaches for machine learning on embedded systems, neuromorphic processing chips are expected to be less power-consuming and overcome the memory bottleneck. In this work, we implemented a pedestrian image detection system on an embedded device using a commercially available neuromorphic chip, NM500, which is based on NeuroMem technology. The NM500 processing time and the power consumption were measured as the number of chips was increased from one to seven, and they were compared to those of a multicore CPU system and a GPU-accelerated embedded system. The results show that NM500 is more efficient in terms of energy required to process data for both learning and classification than the GPU-accelerated system or the multicore CPU system. Additionally, limits and possible improvement of the current NM500 are identified based on the experimental results.

Keywords: embedded system; artificial intelligence; hardware acceleration; neuromorphic processor; power consumption

1. Introduction

An Artificial Neural Network (ANN) consists of a large group of nodes, each of which is assigned a value or synaptic weight to act as an artificial neuron. Calculation of the weight for each neuron for learning and the weighted function value of the input vectors for classification requires large computing power; thus, massively parallel processing can be beneficial. Many-core CPUs and GPUs can be employed in a server for the acceleration of neural network computation to exploit the inherent parallelism of the ANN. Currently, GPUs are the most widely used hardware accelerator of artificial intelligence because GPUs are specialized in performing the same operations on many data instances simultaneously, which is inherently required in the ANN. However, CPUs and GPUs are power-hungry devices, and the energy-intensive computation of the ANN is one of the critical problems that make it difficult for the ANN to be used for power-limited embedded systems.

To make the neural network less power-hungry, the use of specially designed hardware dedicated to neural network performance has been studied. The use of Field-Programmable Gate Arrays (FPGAs) is one such effort; FPGAs consume less energy and can be configured as a custom neural-network-specific hardware [1,2]. A study comparing energy efficiency between an FPGA and GPU in [3] found that simple and parallel computations were performed well on the GPU, but the FPGA outperformed the

GPU in terms of energy efficiency as the complexity of the computational pipeline grew. FPGAs and GPUs offer suitability depending on the application specifications [4]. Thus, the hybrid use of both FPGAs and GPUs [5] has been researched as an efficient implementation of neural networks, especially on embedded systems that require both performance and energy efficiency [6].

Another approach to accelerate the neural network is the ASIC (Application-Specific Integrated Circuit) implementation of neural network models. In particular, neuromorphic chips have been developed to implement brain-like computation to overcome the memory bottleneck problem in parallel processing with von Neumann architecture processors. There are commercial neuromorphic chips available on the market, such as Intel's Loihi and General Vision's NeuroMem. ZISC (Zero Instruction Set Computer) is a hardware implementation of the ANN (Artificial Neural Network) commercialized by IBM, allowing massively parallel processing of digital data [7]. Its feed-forward network provides a nonlinear classifier, which can be used for unknown and uncertainty detection. Based on ZISC technology, General Vision developed the CM1K chip, which consists of 1,024 neurons that can store and process 256-byte length vectors [8,9]. The CM1K chip has been applied to face recognition [9], a fish inspection system [10], and an authentication system by face and speech recognition [8]. The NM500 chip is a successor of CM1K and consists of 576 neurons [11]. Neurons of NM500 have exactly the same behavior as those of CM1K, but it is operated at a higher clock rate and consumes less power. The possibility of adopting the NM500 chip for an ADAS (Advanced Driver Assistance System) has been discussed in [12].

While much research has been done to compare the performance and the energy efficiency of FPGAs and GPUs, little work can be found on neuromorphic chips. IBM's TrueNorth chip is reported as highly energy efficient in [13], and NM500's power consumption is available in the hardware manual. In [14], the authors studied the energy efficiency of a neuromorphic computing system using ReRAM (Resistive RAM). Others compared the performance of neuromorphic computing systems by simulation [15]. However, most research has focused on the neuromorphic chip's performance and power consumption. When it is used as an accelerator in a system, other factors such as data transfer from the host system and control subsystem for interconnection should be considered because the energy cost of data movement is much higher than that of computation [16]. Examining the benefits and the problems of the neuromorphic hardware accelerator for a real-world application needs to include an evaluation of the performance with a real target system.

In this paper, we study the performance and the energy efficiency of a neuromorphic chip employed in an embedded system using a currently available commercial neuromorphic chip, NM500, by comparing its performance and power consumption with those of CPU and GPU cores. To this end, a pedestrian image detection system was implemented and tested on a real target equipped with NM500 chips. The number of neurons in the tested neural network ranges from 576 to 4032 due to the hardware restriction of the evaluation board containing the neuromorphic chips. For these three different configurations (neuromorphic chips, GPUs, and CPUs) of embedded systems, the processing time and power consumption for learning and classification are measured, and the energy efficiency for processing a data instance is calculated and compared.

This paper is organized as follows. Section 2 provides a brief description of the neuromorphic hardware tested in this work. Section 3 explains the datasets and how they are preprocessed for evaluation. In Section 4, experimental results on the power consumption and performance in detecting pedestrian images using NM500 hardware are presented and compared with those on a CPU-only system and a GPU system. Finally, Section 5 discusses applications and possible improvements of the neuromorphic hardware.

2. Neuromorphic Hardware

The neuromorphic chip used in our work to accelerate AI processing in an embedded system is NM500 by Nepes, which is based on the NeuroMem technology of General Vision. An NM500 chip has 576 hardware neurons; a hardware neuron is an identical element that can store and process

information simultaneously. They are all interconnected and working in parallel. NM500 makes these hardware components (neurons) collectively behave as a KNN (K-Nearest Network) or RBF (Radial Basis Function) classifier [11].

Figure 1 shows the interconnection architecture of the NM500. Logically, the network is three-layered: one input layer, one hidden layer, and an output layer. All the neurons in the chip can be considered as nodes in the hidden layer. Each hardware neuron has 256-byte storage, which limits the input size to less than or equal to 256 bytes. The output size is 2 bytes, so the number of candidate class labels for training data is limited to 65,536. Input data and commands are fed to each cell in parallel, and neurons are daisy-chained to signal to the next neuron to accept the input data in training neurons sequentially. Though the layers cannot be made deeper, the hidden layer is extendable by stacking up multiple NM500 chips. The "Neuron interconnect" module shown in Figure 1 enables the use of multiple chips in parallel to expand the size of the neural network by an increment of 576 neurons.

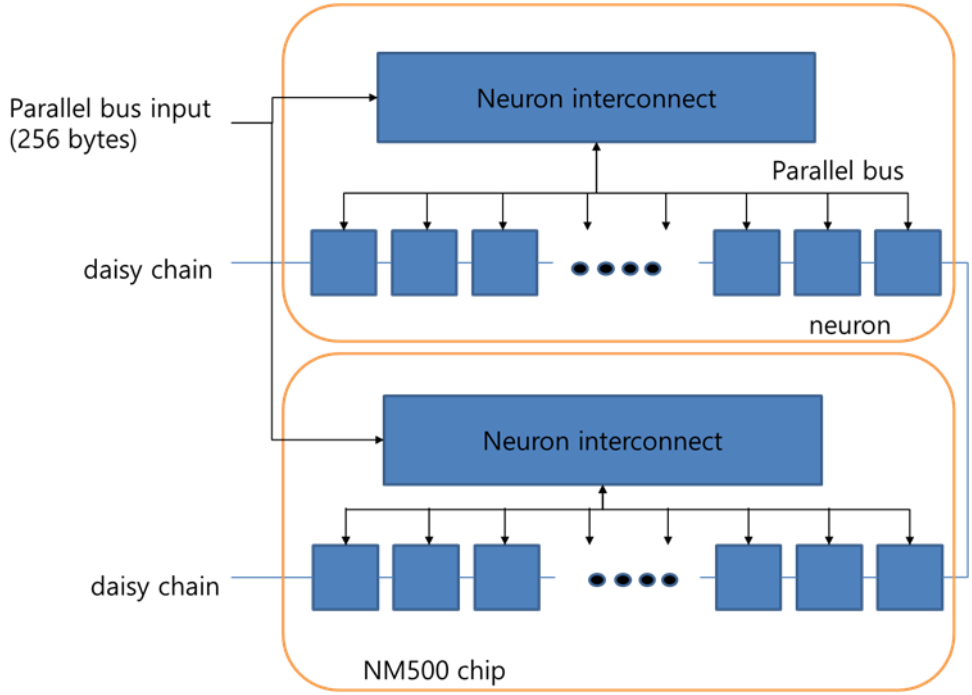

Figure 1. The interconnection architecture of the NM500.

A neuron has a model and IF (Influence Field) value after its learning process, stored in its volatile memory. When data are provided for classification, each neuron calculates the distance of a data point from its model and fires if the distance is less than the IF value. Each neuron examines the response of others on the parallel bus, and if another neuron reports a smaller distance value, then it withdraws itself [11].

The NM500's architecture is not meant to be configured to have a deeper neural network, which may affect accuracy for specific applications. However, because the neural network has only one hidden layer, the classification time of the neuromorphic hardware becomes almost constant. Considering the simplicity of the hardware configuration, it could be suitable for embedded systems whose requirements can be fulfilled by a relatively simple neural network.

3. Benchmark Problem and Data Preprocessing

Because embedded systems have scarce computing resources and power constraints, machine learning problems with a complex model are often infeasible to execute on the system. For example, AlexNet requires about 727 MFLOPS to process a 227 × 227-pixel image, while BCM2835 MCU in Raspberry Pi B development board delivers about 213 MFLOPS at peak operating frequency,

which requires 3.4 s to process an image. Thus, we need to select a machine learning problem that can be feasibly executed on embedded systems.

Pedestrian detection is an important problem in computer vision and has broad application prospects, such as video surveillance, robotics, and automotive safety [17,18]. As autonomous driving systems emerge, detecting small-size pedestrians in images becomes more important [19]. Furthermore, pedestrian detection on an embedded system using GPUs has been studied by researchers, including a recent example in [20]. Thus, because of its practical importance and implementation feasibility, pedestrian detection using the RBF classifier was selected to benchmark the hardware accelerators for embedded systems in our experiments.

For test datasets, the INRIA Person Dataset [21,22], which is very popular in pedestrian detection research [23], was used for our experiments. From the dataset, HOG (Histogram of Oriented Gradient) features with the SVM classifier were extracted to detect a human in [21]. Normalized images in 128-by-64-pixel format were used in our experiments. A pedestrian data point consists of an image pair in which one is a mirror image of another. Non-pedestrian images were generated by randomly selecting 128-by-64-pixel areas from the original image. Table 1 summarizes the images used in our experiments.

Table 1. Dataset for experiments.

Use	Category	No. of Images
Training	Pedestrian	2416
	Non-Pedestrian	2416
Classification	Pedestrian	1132
	Non-Pedestrian	4531

First, the HOG is used as a feature descriptor for human detection to identify a pedestrian. The HOG descriptor methods have shown high performance in human detection and have been widely used for pedestrian detection [24,25]. The image is divided into local regions, where the gradient, direction, and size are computed using the differences in brightness between adjacent pixels. Histograms are generated using the calculated values to make feature vectors.

Because the NM500 chip has a memory of only 256-byte input, the HOG descriptors that are first generated with a high dimension (3780 in our implementation) are not suitable to be used. To reduce the dimension of the features, the PCA (Principal Component Analysis) method was adopted. PCA is a linear transformation feature extraction method that uses less data than the input data while maintaining the most important information of the input data [25,26]. Using the PCA method, the HOG features are transformed into 40-dimensional data. Finally, 32-bit floating-point features are quantized to 8-bit data using the vector quantization method in [27], which results in 160-byte input vectors.

4. Experimental Results

The neuromorphic chips are packaged into a hardware module named NeuroShield. The NeuroShield module is an evaluation board with one NM500 chip containing 576 neurons, on top of which three more extension modules can be stacked at most. The extension module has two NM500 chips and is called NeuroBrick. Thus, a NeuroShield module supports at most 4032 neurons. Because the NeuroShield module has one NM500, and each NeuroBrick module has two NM500 chips, the neuromorphic processing can be done with four different numbers of neurons: 576 with only the NeuroShield module, 1728 with one NeuroBrick on it, 2880 with two NeuroBricks, and 4032 at maximum.

To compare the neuromorphic chip-based system with other systems, the same pedestrian detection neural network was implemented on a CPU-only system and a GPU system. The CPU-only system is an embedded board with the Exynos5422 processor. The Exynos5422 CPU of the embedded board used in our experiments has an 8-core CPU, which consists of 4 fast (big) cores and 4 slow (little)

cores. Because little cores are too slow to be used for neural network processing, only big cores were used in the experiments.

As a GPU system for comparison, Nvidia's Jetson Nano board with 128 GPU cores was used. The neural network on the GPU system was implemented using the Tensorflow-GPU library. The NeuroShield module was connected to the Jetson Nano board using an SPI (Serial Peripheral Interface) connection at 2 MHz. The power consumption of the NeuroShield module was measured separately from the Jetson Nano board. Table 2 summarizes the hardware configurations of each system test in the experiments. The neuromorphic system (NeuroShield) does not have a CPU or external memory because NM500 is not a Von Neumann architecture.

Table 2. Hardware specifications.

System Type	Accelerator	CPU	Memory
CPU system	None	Octa-core ARM @2 GHz (A15 × 4 & A7 × 4)	2 GB LPDDR4
GPU system	GPU (128 cores)	Quad-core ARM @1.43 GHz A57 × 4	4 GB LPDDR4
Neuromorphic system	NM500 (up to 7 chips)	-	256 B per each neuron

Figure 2 shows the power consumption of three implementations with different underlying hardware: NM500, GPU, and CPU-only. For the CPU-only system, we measured the power consumption in two cases: using only one core and using all four big cores. The GPU system consumes about 4.85–4.89 Watts on average. The power consumption of the GPU system includes the power consumed by the CPU in the system. For a fair comparison with the neuromorphic hardware, the power consumed only by the GPU cores needs to be measured, which could not be done because the GPU cores are integrated with CPU cores in the Jetson Nano SoC. Therefore, to estimate the power consumption of the GPU cores, the power consumption of the board with GPU cores off at idle state was measured, which is about 1.41 Watts on average. The GPU line in Figure 2 shows the power consumption of the entire system, while the GPU–idle line shows the power consumption estimated by subtracting the average power consumption of the Jetson Nano board at idle state with the GPU cores off.

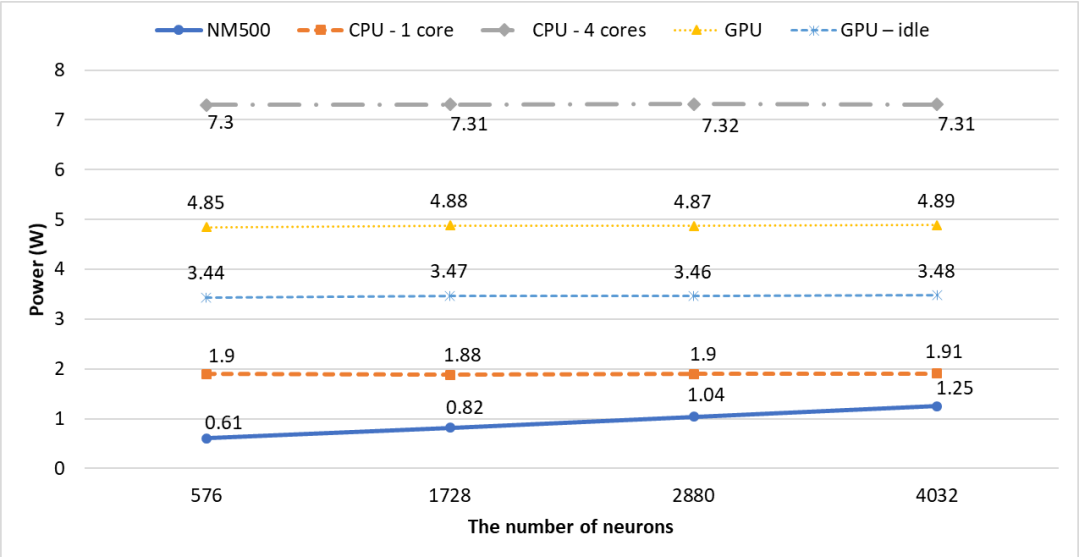

Figure 2. Average power consumption for varying numbers of neurons.

The GPU cores cannot be turned off partially; thus, the power consumption remains the same regardless of the number of neurons. The power consumptions of both the CPU-only and GPU systems are barely changed, while that of the NM500 system linearly increases as the number of chips (hardware neurons) employed increases. An additional NM500 chip containing 576 hardware neurons consumes approximately 100mW, so the total power consumption increases by 0.2 W for every additional 1172 neurons. The power consumption of the NeuroShield is a little higher than that of the NeuroBrick because the NeuroShield has an FPGA for interfacing with SPI, I2C, and USB.

Figure 3 shows the amount of time needed to train a neural network for the three systems. The time in the graph is in a base-10 log scale. Training on the CPU-only system takes much longer than other systems, about 15–185 times longer than the GPU system even though four cores are used. The learning time of the neuromorphic chip-based system is even shorter than that of the GPU system, by about 1300–1500%.

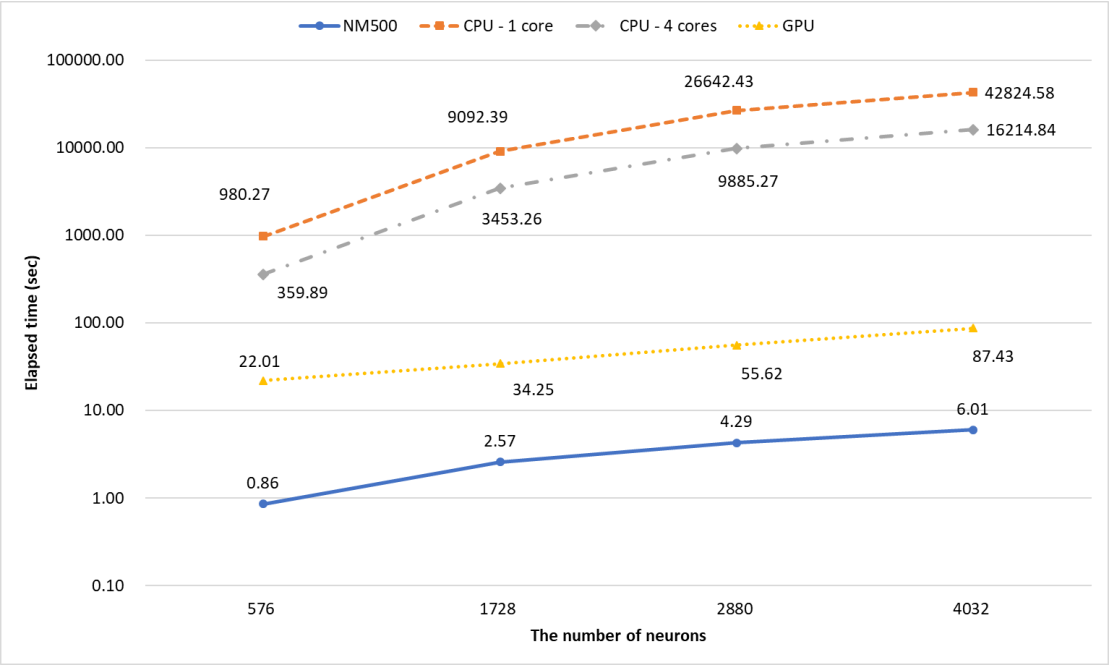

Figure 3. Time to train the model.

To determine the classification performance, 1132 pedestrian images and 453 non-pedestrian images were tested. The processing time of the neuromorphic-based system is unchanged even though the number of neurons used in the system is increased, as shown in Figure 4, because all hardware neurons are in the same layer and execute in parallel. On the other hand, the processing times of the CPU-only system and the GPU system increase as the number of neurons increase. Nevertheless, the GPU system is the fastest among the systems to process all the test images. We could not test with more neurons because the current NeuroShield does not support more than three NeuroBricks on it.

To determine the energy efficiency of the neuromorphic hardware, we calculated the amount of energy needed to process an image when each system is learning or classifying. The consumed energy for learning is shown in Figure 5, while the energy for image detection is shown in Figure 6.

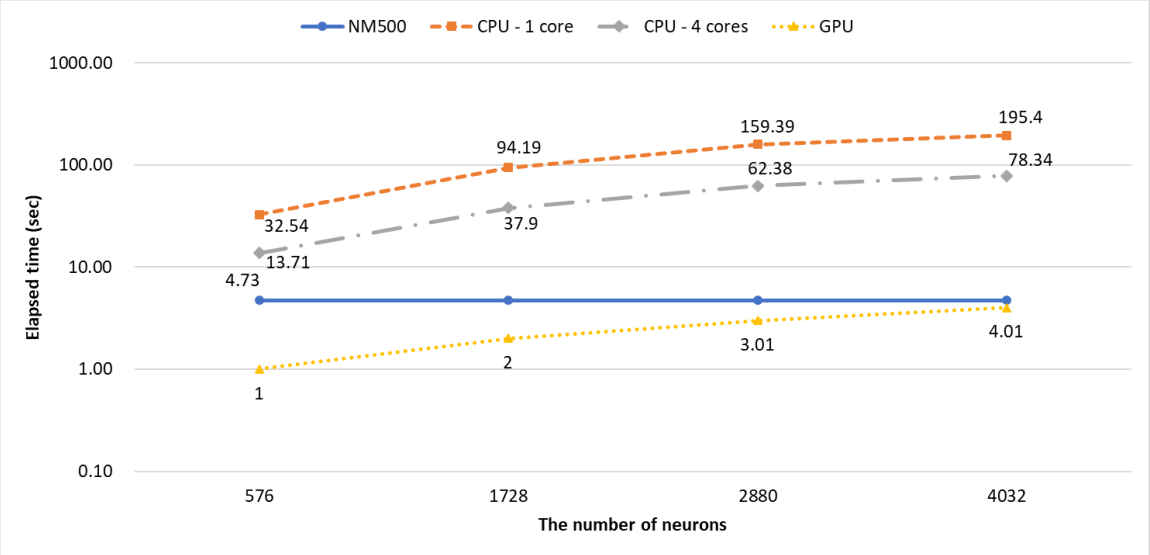

Figure 4. Time for processing all test data.

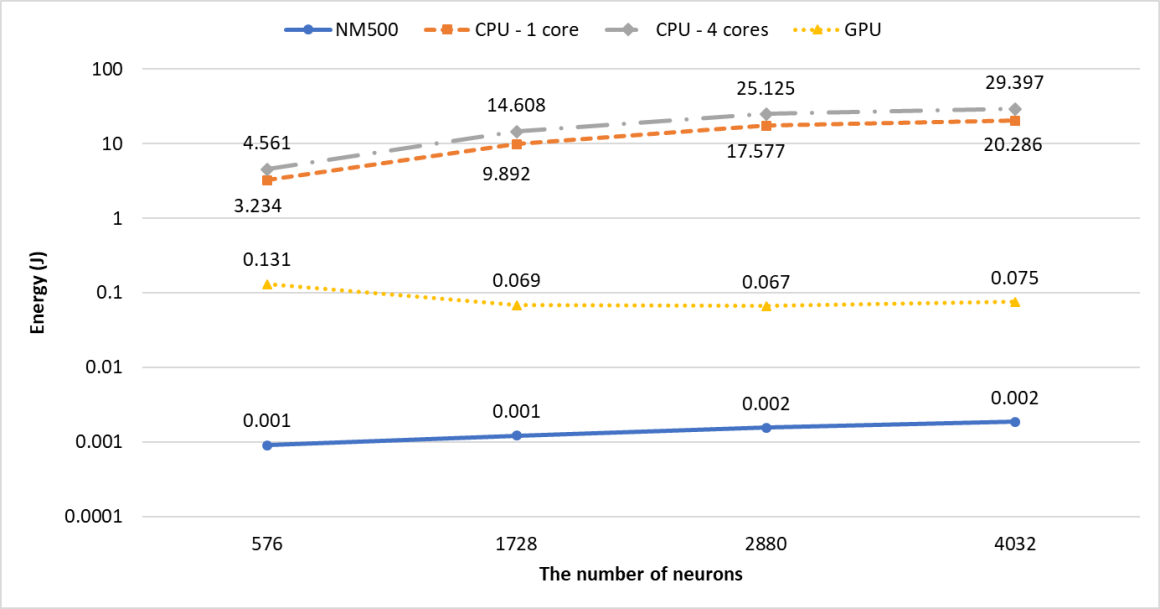

Figure 5. Energy per training data.

When training the neural network, the system with NM500 requires the least amount of energy to process an image data instance. The GPU system's energy efficiency is much better than that of the CPU-only system, but the efficiency of the neuromorphic chip-based system is over an order of magnitude higher than that of the GPU system in terms of per-data energy consumption. This is because both the power consumption and the processing time of the NM500 are much lower than those of the GPU system.

For classification, the required amount of energy consumed for a data instance by the GPU system is more than that of the neuromorphic system but is not very high (<2.5 times). Though the power consumption of the GPU system is higher than that of the NM500 system, the processing time of the GPU system is shorter than that of the NM500 system. The energy consumption of the GPU system in classification is much lower than that in training because less computation is needed in classification. However, the NM500 system consumes about 80% more energy for classification than learning.

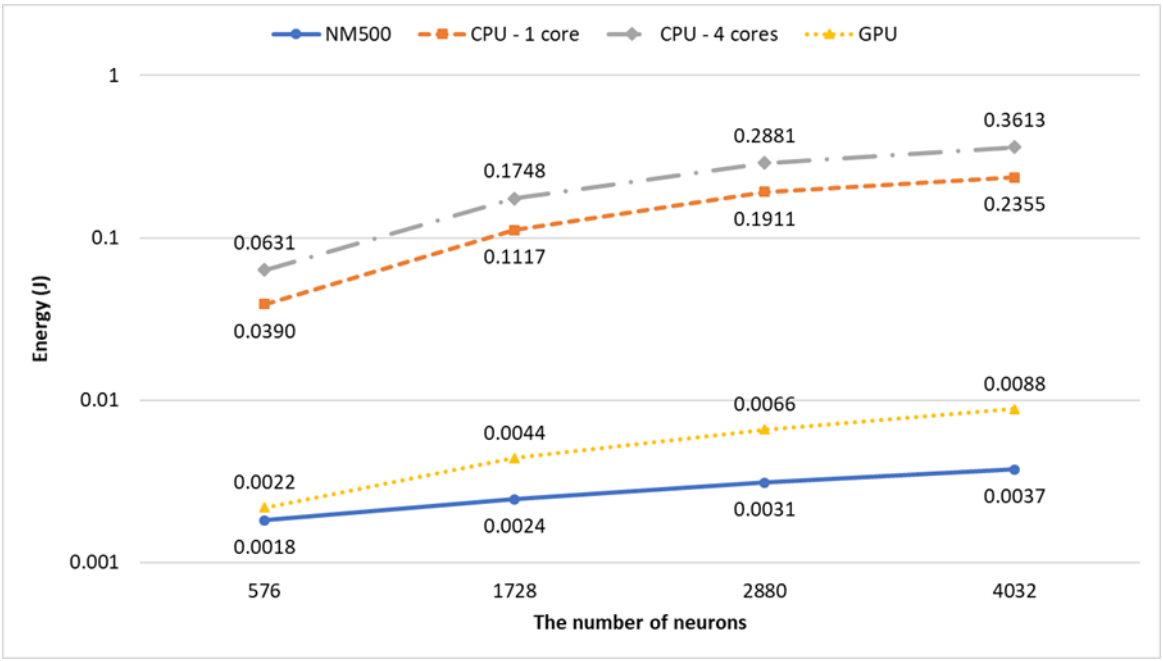

Figure 6. Energy per test data.

5. Conclusions

In this work, we compared the performance and the energy efficiency of a commercially available neuromorphic chip with those of GPU and CPU cores on embedded devices. The processing time and power consumption were measured while increasing the number of neurons in the neural network. The experimental results show that the time required for the neuromorphic chips to learn the same amount of data is about 13–15 times shorter than that required for the embedded system with 128 GPU cores. On the other hand, the time required to classify a dataset remains almost constant for the neuromorphic processor due to its neural network architecture with only one hidden layer. In classification, the GPU processes data faster than the neuromorphic chip, but the processing time tends to increase as the number of neurons increases. Thus, it is expected that the neuromorphic chips can outperform the GPU system with a larger number of neurons, but this could not be tested due to the restriction in the expandability of the evaluation board for the neuromorphic chips.

As most embedded systems depend on limited power supply such as batteries, energy efficiency is a critical factor in designing a system. In our experiments, the energy required for NM500 chips to process an input data instance is less than 1/35 of that required by the GPU accelerated embedded system in training the neural network, while the energy consumption of the GPU system in classification is only 1.22–2.37 times higher than that of NM500 chips. Because the neural network of NM500 has only one hidden layer, the processing time for classifying the given dataset remains almost the same even though the number of neurons increases, while the power consumption increases linearly. On the contrary, the classification time of the GPU system is almost linearly proportional to the number of neurons, while the power consumption remains unchanged. Therefore, the energy for the neuromorphic chips to classify data is expected to be close to 50% of that for the GPU system as the number of chips increases. It is interesting to note that the neuromorphic chips are especially energy efficient in training the neural network in comparison with the GPU system.

However, despite the high energy efficiency of the neuromorphic chip, it still needs to be improved. First of all, NM500's architecture can support only one hidden layer, which may restrict the possible benefit of deeper neural networks. Second, the lack of the ability to dynamically switch on and off a partial group of chips makes it difficult to manage power consumption. Finally, if high-speed interconnection such as AMBA or other high-speed bus protocols is supported, the processing time

can be reduced further. The current SPI connection of the NM500 requires about 10 us to write a byte, a total 1660 us for learning a 160-byte data instance, which is done in about 2045 us. Therefore, communication via SPI interconnection accounts for about 82% of the total processing time. Integration of the neuromorphic chips into an SoC could be considered to improve the performance and the energy efficiency of machine learning on embedded systems using the neuromorphic hardware.

Author Contributions: M.K. performed most of the experiments in this paper, Y.L. performed preprocessing and analysis, M.P. performed analysis and wrote the original draft preparation. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work was supported by Incheon National University Research Fund in 2019.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Guo, K.; Zeng, S.; Yu, J.; Wang, Y.; Yang, H. A survey of FPGA-based neural network interface accelerator. *ACM Trans. Reconfig. Technol. Syst.* **2018**, *12*, 2.
2. Shawahna, A.; Sait, S.M.; El-Maleh, A. FPGA-based accelerators of deep learning networks for learning and classification: A review. *IEEE Access* **2019**, *7*, 7823–7859. [CrossRef]
3. Qasaimeh, M.; Denolf, K.; Lo, J.; Vissers, K.; Zambreno, J.; Jones, P.H. Comparing energy efficiency of CPU, GPU and FPGA implementations for vision kernels. In Proceedings of the IEEE International Conference on Embedded Software and Systems, Las Vegas, NV, USA, 2–3 June 2019; pp. 1–8.
4. Jahnke, M.D.; Cosco, F.; Novickis, R.; Rastelli, J.P.; Gomez-Garay, V. Efficient neural network implementations on parallel embedded platforms applied to real-Time torque-vectoring optimization using predictions for multi-motor electric vehicles. *Electronics* **2019**, *8*, 250. [CrossRef]
5. Liu, X.; Ounifi, H.A.; Gherbi, A.; Lemieux, Y.; Li, W. A hybrid GPU-FPGA-based computing platform for machine learning. *Procedia Comput. Sci.* **2018**, *141*, 104–111. [CrossRef]
6. Tu, Y.; Sadiq, S.; Tao, Y.; Shyu, M.; Chen, S. A power efficient neural network implementation on heterogeneous FPGA and GPU devices. In Proceedings of the IEEE 20th International Conference on Information Reuse and Integration for Data Science, Los Angeles, CA, USA, 30 July–1 August 2019; pp. 193–199.
7. Zhang, D.; Ghobakhlou, A.; Kasabov, N. An adaptive model of person identification combining speech and image information. In Proceedings of the 8th Control, Automation, Robotics and Vision Conference, Kunming, China, 6–9 December 2004; pp. 413–418.
8. Suri, M.; Parmar, V.; Singla, A.; Malviya, R.; Nair, S. Neuromorphic hardware accelerated adaptive authentication system. In Proceedings of the IEEE Symposium Series on Computational Intelligence, Cape Town, South Africa, 7–10 December 2015; pp. 1206–1213.
9. Sardar, S.; Tewari, G.; Babu, K.A. A hardware/software co-design model for face recognition using Cognimem Neural Network chip. In Proceedings of the International Conference on Image Information Processing, Himachal Pradesh, India, 3–5 November 2011.
10. Menendez, A.; Paillet, G. Fish inspection system using a parallel neural network chip and the image knowledge builder application. *AI Mag.* **2008**, *1*, 21–28.
11. General Vision. NeuroMem Technology Reference Guide, Version 5.4. Available online: https://www.general-vision.com/documentation/TM_NeuroMem_Technology_Reference_Guide.pdf (accessed on 30 June 2020).
12. Kim, J. New neuromorphic AI NM500 and its ADAS application. In Proceedings of the International Conference on Advanced Engineering Theory and Applications, Ostrava City, Czech Republic, 11–13 September 2018; pp. 3–12.
13. Esser, S.K.; Merolla, P.A.; Arthur, J.V.; Cassidy, A.S.; Appuswamy, R.; Andreopoulos, A.; Berg, D.J.; McKinstry, J.L.; Melano, T.; Barch, D.R.; et al. Convolutional networks for fast, energy-efficient neuromorphic computing. *Proc. Natl. Acad. Sci. USA* **2016**, *113*, 11441–11446. [CrossRef] [PubMed]
14. Li, P.; Yang, C.; Chen, W.; Huang, J.; Wei, W.; Liu, J. A neuromorphic computing system for bitwise neural networks based on ReRAM synaptic array. In Proceedings of the IEEE Biomedical Circuits and Systems Conference, Cleveland, OH, USA, 17–19 October 2018; pp. 1–4.

15. Zhao, Z.; Wang, Y.; Zhang, X.; Cui, X.; Huang, R. An energy-efficient Computing-In-Memory neuromorphic system with on-chip Training. In Proceedings of the IEEE Biomedical Circuits and Systems Conference, Nara, Japan, 17–19 October 2019; pp. 1–4.
16. Sze, V.; Chen, Y.-H.; Yang, T.-J.; Emer, J.S. Efficient processing of deep neural networks: A tutorial and survey. *Proc. IEEE* **2017**, *105*, 2295–2329. [CrossRef]
17. Dollar, P.; Wojek, C.; Schiele, B.; Perona, P. Pedestrian detection: A benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 304–311.
18. Dollar, P.; Wojek, C.; Schiele, B.; Perona, P. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 743–761. [CrossRef]
19. Zhang, X.; Cao, S.; Chen, C. Scale-aware hierarchical detection network for pedestrian detection. *IEEE Access* **2020**, *8*, 94429–94439. [CrossRef]
20. Barba-Guaman, L.; Naranjo, J.E.; Ortiz, A. Deep learning framework for vehicle and pedestrian detection in rural roads on an embedded GPU. *Electronics* **2020**, *9*, 589. [CrossRef]
21. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–26 June 2005; pp. 886–893.
22. INRIA Person Dataset. Available online: https://dbcollection.readthedocs.io/en/latest/datasets/inria_ped.html (accessed on 24 February 2020).
23. Taiana, M.; Nascimento, J.C.; Bernardino, A. An improved labelling for the INRIA person data set for pedestrian detection. *Lect. Notes Comput. Sci.* **2013**, *7887*, 286–295.
24. Zhu, Q.; Avidan, S.; Yeh, M.; Cheng, K. Fast human detection using a cascade of histograms of oriented gradients. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–25 June 2006; pp. 1491–1498.
25. Kim, J.-Y.; Park, C.-J.; Oh, S.-K. Design & Implementation of Pedestrian Detection System Using HOG-PCA Based pRBFNNs Pattern Classifier. *Trans. Korean Inst. Electr. Eng.* **2015**, *64*, 1064–1073.
26. Jiang, J.; Xiong, H. Fast pedestrian detection based on HOG-PCA and gentle AdaBoost. In Proceedings of the 2012 International Conference on Computer Science and Service System, Nanjing, China, 11–13 August 2012; pp. 1819–1822.
27. Benoit, J.; Skirmanta, K.; Bo, C.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; Kalendichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2704–2713.