UDK 004.875; 004.97
DOI: 10.31891/CSIT-2021-3-9

YELYZAVETA HNATCHUK, YEVHENIY SIERHIEIEV, ALINA HNATCHUK
Khmelnytskyi National University, Khmelnytskyi, Ukraine

USING ARTIFICIAL INTELLIGENCE ACCELERATORS TO TRAIN COMPUTER GAME CHARACTERS

A review of the literature has shown that today, given the complexity of computational processes and the high cost of these processes, the gaming computer industry needs to improve hardware and software to increase the efficiency and speed of processing artificial intelligence algorithms. An analysis of existing machine learning tools and existing hardware solutions to accelerate artificial intelligence. A reasonable choice of hardware solutions that are most effective for the implementation of the task. Possibilities of practical use of the artificial intelligence accelerator are investigated. The effectiveness of the proposed solutions has been proven by experiments. The use of an artificial intelligence accelerator model allowed to accelerate the learning of a computer game character by 2.14 times compared to classical methods.

Keywords: processor, artificial intelligence, computer games, agent, machine learning.

ЄЛИЗАВЕТА ГНАТЧУК, ЄВГЕНІЙ СЄРГЄЄВ, АЛІНА ГНАТЧУК
Хмельницький національний університет, Хмельницький, Україна.

ВИКОРИСТАННЯ ПРИСКОРЮВАЧІВ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ НАВЧАННЯ ПЕРСОНАЖІВ КОМП'ЮТЕРНИХ ІГОР

В статті проаналізовано існуючі інструментальні засоби для полегшення та пришвидшення процесу розробки ігор. Однією з головних проблем, з якими стикаються розробники таких систем та додатків, є те, що алгоритми штучного інтелекту є обчислювально дорогими. Цей факт змушує шукати шляхи ефективного вирішення цієї проблеми, зокрема використання поліпшення апаратного прискорення, щоб забезпечити необхідну високу обчислювальну потужність. Оптимізована та спеціалізована апаратна реалізація може зменшити системні витрати за рахунок оптимізації необхідних ресурсів та зменшення вимог до енергії при одночасному поліпшенні продуктивності. Все більше дослідників, зокрема зарубіжних, досліджують галузь та пропонують нові інструментальні засоби для полегшення та пришвидшення процесу розробки ігор.

Враховуючи складність обчислювальних процесів та велику вартість цих процесів, ігрова комп'ютерна галузь потребує вдосконалення апаратного та програмного забезпечення для підвищення ефективності та швидкості обробки алгоритмів штучного інтелекту. Доведено, що використання прискорювачів штучного інтелекту для навчання персонажів комп'ютерних ігор з використанням методів машинного навчання є актуальною задачею. Це дозволяє підвищити ефективність та швидкість обробки алгоритмів штучного інтелекту. Проведений аналіз показав, що для конкретної задачі, що розглядається в даній роботі, доцільним для прискорення навчання ШІ є використання моделі прискорювача штучного інтелекту, що використовує процесор Intel I9 11900 (11 покоління), який має нову архітектуру для штучного інтелекту – Intel Deep Learning Boost. Досліджено можливості практичного використання прискорювача штучного інтелекту. Ефективність використання запропонованих рішень доведено експериментами. Використання моделі прискорювача штучного інтелекту дозволило в 2,14 рази пришвидшити навчання персонажу комп'ютерної гри порівняно з класичними методами.

Ключові слова: процесор, штучний інтелект, комп'ютерні ігри, агент, машинне навчання.

Introduction

Artificial intelligence (AI) and machine learning tools have become widespread in recent years, thanks to advances in computing in power, computing, and performance.

Artificial intelligence is used in an extremely wide range of programs to get better results compared to traditional methods.

Such applications include image processing, such as face detection and recognition [1], financial markets analysis and banking [2], robotic systems in industry [3], medical applications and healthcare applications [4], efficient transactions in database management [5], security applications [6], unmanned delivery services and personal transport [7], autonomous drones for navigation [8] and much more.

Given the requirements of such applications for accuracy and efficiency, many of these applications are based on artificial intelligence. This trend indicates the constant interest and high potential of artificial intelligence tools and machine learning.

These tools are increasingly becoming an integral part of every electronic or embedded system. Microrobots with built-in artificial intelligence are widely used in practice in various fields. The robots include advanced controllers that implement AI functions, such as perception (information retrieval) and cognition (decision making). One of the main problems faced by developers of such systems and applications is that artificial intelligence algorithms are computationally expensive. This fact leads to the search for ways to effectively solve this problem, in particular the use of improved hardware acceleration to provide the necessary high computing power.

Optimized and specialized hardware implementation can reduce system costs by optimizing the required resources and reducing energy requirements while improving productivity [9].

Related works

In [11] a study of the hardware implementation of artificial intelligence algorithms and machine learning from 2009 to 2019. The main purpose of this work was to introduce neural networks as a tool for detecting and recognizing objects in various applications. In this study, the results of two hundred and one scientific works are analyzed and presented, 169 works were related to the hardware implementation of algorithms of artificial intelligence and machine learning. Hardware acceleration is considered an ideal solution for energy-intensive and resource-intensive artificial intelligence algorithms.

The aim is to achieve faster and more efficient processing of AI algorithms [12]. In recent years, many studies have been published that discuss a large number of implementations of hardware and software optimization and implementation methods in this area [13-20]. Part of the research looked at the implementation of artificial neural networks in hardware in general, other studies focused on FPGA accelerators for deep learning neural networks. For example, in [15] the authors focused on the implementation of FPGA wrapped neural networks. In [17], the study discussed the details of the implementation of the GPU.

In [21] all the main hardware implementations of artificial neural networks implemented on hardware, called hardware neural networks (ASM), were studied. Hardware neural networks are classified according to some characteristics, such as embedded / disabled chip, analog / digital blocks, thresholds, lookup table, calculation and data rate. A special section for ASM chips covers the implementation of digital and hybrid neurochips based on FPGA and ASIC.

In [22] a more detailed study was conducted, the main aspects of which were devoted to neural network accelerators based on FPGA, where they investigated the design of neural network accelerators and summarized all the methods used to automate the design of accelerators.

The study [24] discussed the basics of learning deep neural networks with this section for methods of implementing compression. The survey analyzes FPGA-based accelerators and ASIC-based accelerators, with a greater emphasis on FPGAs. Other studies, such as [25], have also considered techniques for designing FPGA-based accelerators for compression neural networks.

In conclusion, we can say that all studies have examined various alternatives to the GPU in a comprehensive way to improve the performance of artificial intelligence algorithms. The use of GPUs is a good option in terms of accelerating artificial intelligence and an option that can compete with solutions based on FGPA and ASIC.

A review of the literature has shown that the popularity of computer games is growing today. More and more researchers, including foreign ones, are researching the industry and offering new tools to facilitate and speed up the game development process.

Given the complexity of computational processes and the high cost of these processes, the gaming computer industry needs to improve hardware and software to increase the efficiency and speed of processing artificial intelligence algorithms.

Therefore, the use of artificial intelligence accelerators to train computer game characters based on machine learning techniques is an urgent task.

Using microcomputers for machine learning

An important issue is the choice of technology that increases the productivity and efficiency of artificial intelligence in games. Interesting for practical use is Intel's technology designed to automatically increase the clock speed of the processor above the nominal, which is called Turbo Boost. However, the power, temperature and current limits in the design capacity must not be exceeded. The use of such technology increases the performance of single-stream and multi-stream programs.

In other words, it is actually a processor overclocking technology. The availability of Turbo Boost technology does not depend on the number of active cores. It depends on the presence of one or more cores that can operate at a power that is below the design. The operating time of the system in this technology depends on such things as workload, platform design and operating conditions. Intel Turbo Boost technology is enabled in one of the BIOS menus by default.

Another technology is a neural processor or artificial intelligence accelerator (NPU). This is a specialized class of microprocessors or otherwise coprocessors, which is most often used to hardware accelerate the work of artificial intelligence. Such as artificial neural network algorithms, voice recognition, computer vision, machine learning and other artificial intelligence methods. Neural processors are computer technology and are used to hardware accelerate the emulation of neural networks, as well as real-time digital signal processing.

Most often, neural processors include store-type memory blocks, registers, a computing device containing a multiplication matrix, a switch, decoders, multiplexers, and triggers.

The class of neural processors includes:

- Neuromorphic processors built on a cluster architecture. Unlike traditional computing architectures, they are highly specialized for the creation and development of various types of artificial neural networks. These processors use conventional transistors. Computational cores are built from them. Each kernel contains a router to communicate with other kernels, a task scheduler, and its own SRAM. Each nucleus emulates the work of a large number of neurons. Such processors are used for deep machine learning.

- Tensor processors, which are coprocessors, are controlled by a CPU operating with tensors. They have their own built-in RAM and are highly specialized for performing matrix multiplication and convolution operations. These operations are used to emulate wrapped neural networks used for machine learning.
- Machine vision processors have many similarities with tensor processors. But they differ from them in that they are used to accelerate the operation of machine vision algorithms, which use the methods of convolutional networks, as well as scale-invariant transformation of features. The emphasis is on parallelizing the flow of data between multiple executive cores. They, like tensors, are used for low-precision calculations, which is accepted in image processing.

In the case of machine learning using microcomputers, note that the reinforced learning model cannot learn on a RaspberryPi or using another microcomputer because their boards do not have enough power to perform a large number of operations. During training, a large number of floating-point multiplication operations take up all the computing resources of the microcomputer, so it is only possible to import and run a ready-made model on microcomputers.

Let's consider the basic boards of microcomputers, their parameters and possibilities of additional modules for machine learning. In [15] the review of comparison of efficiency of microcomputer platforms of machine learning is given. The review concluded that the most affordable solution for learning at an affordable price would be a combination of a Raspberry Pi 4 microcomputer with an Intel Neural Compute Stick 2 and a TensorFlow framework.

TensorFlow from Google is the open and most popular deep learning library for research and development. TensorFlow is a cross-cutting platform that simplifies the creation and deployment of ML models. TensorFlow offers several levels of abstraction so you can choose the right one for your needs. You can create and train models using the high-level Keras API, which makes it easy to get started with TensorFlow and machine learning. If more flexibility is required, persistent execution allows immediate iteration and intuitive debugging.

For large machine learning tasks, distribution strategy APIs are used for distributed learning on different hardware configurations without changing the model definition.

An important issue in the calculation of artificial intelligence in games and the use of machine learning is acceleration, which allows you to effectively address the allocation of CPU resources and reduce the learning time of artificial intelligence.

Consider the existing and most commonly used hardware accelerators of artificial intelligence and machine learning, based on standard quality indicators. Hardware accelerator documents are classified from a hardware perspective to provide a useful comparison. These are the five categories listed below and shown in Figure 1:

- hardware mapping;
- accelerator design based on FPGA;
- accelerator design based on GPU;
- reclassification of accelerators;
- Xeon-Phi accelerator design.

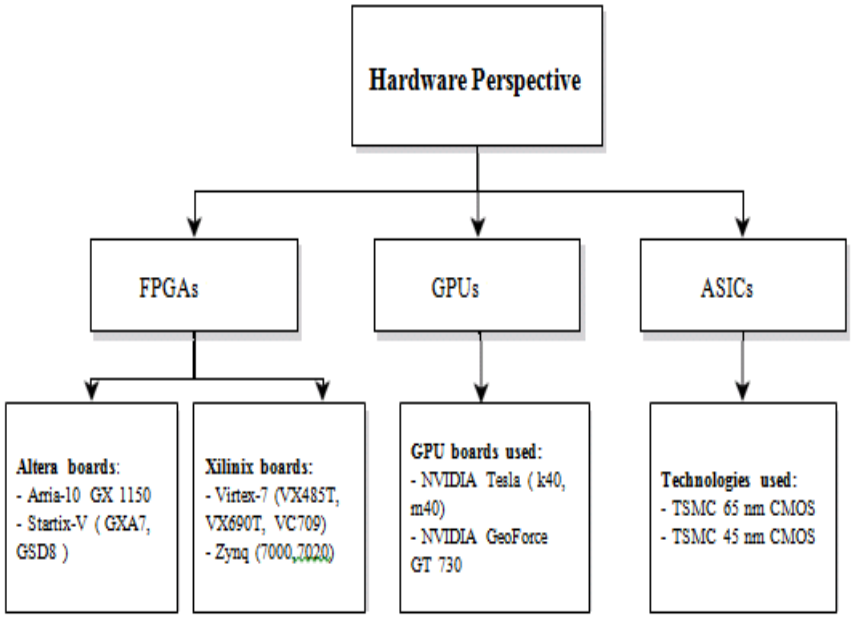

Fig. 1 Hardware accelerators of artificial intelligence

Consider in more detail the scope of each of them. The problem facing researchers is to find an appropriate algorithm for a particular application and its effective application to the hardware. CNNs are built using direct transmission networks, which can be considered as a large number of similar scalar operations transmitted at different stages. This type of calculation is significantly accelerated when working on FPGA. FPGA mapping tasks include finding an effective match between the CNN computing model and the execution model supported by the FPGA. Several research papers have addressed these issues of CNN mapping on FPGA.

The heuristic search algorithm and data stream command set architecture used DNNWEAVER to create high-performance accelerators running on a limited energy budget and built-in FPGA memory. DNNWEAVER is a structure that automatically generates a synthesized accelerator for a given pair (DNN, FPGA) using manually optimized templates. The Deep Burning structure is built to simplify the operation of displaying various neural networks in FPGA or ASIC. RTL library compiler for automatic creation of individual FPGA-based accelerators for this AI algorithm.

The possibility of direct hardware mapping of AI on FPGA is considered by various researchers. The proposed model states that it opens up new opportunities for further optimization and can be extended to ASIC technology, as well as binary neural networks. There is a delay-based design methodology for mapping ConvNets to FPGAs. Python-based automation tools for creating FPGA-based AI accelerators are quite popular to use. The tool constantly provides high throughput for various AI models.

The RTL library compiler achieved the best performance: 732.36 GOPS, 1604.57 GOPS, 651.49 GOPS and 789.44 for NiN, VGG-16, ResNet-50 and ResNet-152 on Startix 10 GX2800 FPGA respectively. For AlexNet, the python-based automation tool showed the best performance, which is 274.5 GOPS.

Many FPGA-based accelerator architecture designs have been proposed. Most of the research work was based on FNGA-based CNN accelerators. From a hardware point of view, these documents are divided into two categories:

- heterogeneous architecture;
- FPGA.

Heterogeneous architecture accelerators use a high-quality FPGA processor to implement a high-performance binary neural accelerator (BNN). The proposed accelerator is designed to take advantage of the Xeon CPU + FPGA system. The FPGA architecture is designed for the most computational parts of the BNN, while other parts of the topology can be processed by the CPU. There are other AI accelerators based on CPU-FPGA. Intensive computational and universal operations (eg, 2D convolutions) were implemented in the FPGA.

Other heterogeneous architectures combining ASIC with FPGA have focused on deep learning with efficient tensor matrix / vector operations. The application of the accelerator architecture using two FPGAs is promising. Implementation of the accelerator based on AI FPGA reached an accuracy of 82.8%.

Xilinix FPGA-based AI accelerators have the highest performance among other implementations. Their implementation provides the latest results. This is due to the use of interlayer planning, which has reduced data transfer from 6.6 MB to 0.2 MB with a reduction of 97%.

FPGA-based AI accelerators show the highest performance. This architecture achieves high performance due to several sources of parallelism integrated into the implementation of AI, which aim to achieve the best possible acceleration.

Other FPGA-based AI accelerators have been implemented using altera boards. As for the implementation with a fixed point, it is known about the highest efficiency among other implementations. They addressed the performance constraints caused by the low bandwidth of the built-in memory due to the two-dimensional relationship between the processor elements (CPUs) and the local memory. This effectively increases the effective bandwidth of the built-in memory. Whereas for 32 floating-point implementations on altera boards, the highest performance is reported.

The methods used to achieve this high performance were to use a built-in flow buffer that efficiently stores input and output function maps. In addition, a vectorization approach was used, which achieves more than 60% DSP efficiency.

Built-in FPGA processors, such as NIOS-II, and ARC processors in SoC-based FPGAs are also used to implement the target accelerators. AI acceleration can also be achieved by using parallel programmable logic to implement convolution, merging, and add-ons, as well as a built-in ARM processor to run and perform other tasks.

GPUs are becoming important to accelerate deep learning. Existing methods are aimed at accelerating algorithms on GPUs.

Over the past decade, Intel Xeon Phi coprocessor has paved the way for success in implementing deep learning algorithms. Intel Xeon Phi is a shared memory. This means a multi-core coprocessor that contains up to 61, 1.2 GHz cores, and each core can switch between 4 hardware threads in a circular manner. Xeon Phi makes extensive use of Single Instruction Multiple Data (SIMD) technology, which allows you to perform the same operation on multiple data segments simultaneously. Thus, Xeon Phi is very suitable for deep learning programs, which usually use simple operations on large tensors.

Researchers have shown that Intel Xeon Phi can offer an efficient but more general way to parallelize the deep learning algorithm compared to GPUs. In-depth learning can be effectively optimized and scaled on multi-core HPC systems.

The proposed CosmowFlow, the first large-scale scientific application of the TensorFlow framework on a supercomputer scale, is completely synchronous. Optimized full software stack that includes network design, I / O processing, communication, TensorFlow framework, and 3D CNN placement in MKLDNN for 3D convolutional neural network. Some studies by mid-level researchers on BNN have proposed a new approach to optimizing BNN on processors using the BitFlow framework BitFlow gets 1.8 accelerations.

Hardware technologies for machine learning:

- programmable FP-DNN, a structure having the input data of the described TensorFlow DNN, used to generate hardware implementations on FPGA boards with hybrid RTL-HLS templates;
- structure used to automate the display of AI on systolic arrays on FPGA;
- TuRF AI acceleration structure, inspired by efficient AI architectures and transfer training, which supports optimization for a specific domain. It efficiently uses domain programs on the FPGA;
- Caffe uses an open source deep learning system to provide clear access to deep architectures. The code is written in pure, efficient C ++, where CUDA is used to calculate the GPU;
- clCaffe, an acceleration of OpenCL's well-known Caffe deep learning mechanism, while focusing on the convolution level, which has been optimized by three different approaches. This greatly improves the ability to use deep learning cases on all types of OpenCL devices.

There are many open source tools supported by NVIDIA and Intel to support accelerating machine learning algorithms. For example, NIVIDIA has developed RAPIDS, a machine learning library with an accelerated graphics processor, which aims to significantly speed up the learning process. RAPIDS improves performance by accelerating the complete pipeline pipeline, including data preparation, machine learning algorithm, and GPU visualization.

RAPIDS also significantly speeds up processing, trains larger datasets, and supports the deployment of multiple GPUs. Fast is used to successfully reduce the required training time of systems that recommend what is computationally considered intensive with an acceleration of 15.6 times.

The analysis showed that for the specific task considered in this paper, it is advisable to accelerate the learning of AI is to use a model of artificial intelligence accelerator using Intel I9 11900 processor (11th generation), which had a new architecture for artificial intelligence - Intel Deep Learning Boost .

Experiments

To prove the effectiveness of the use of artificial intelligence accelerators in computer games, an experiment was conducted to train a shock-blocking agent. This paper discusses the following key mechanics of a computer game. These are fist fights. If you omit the means of movement and orientation, the player will have access to 8 actions and 8 pathogens from which the decision will be made on the following moves:

1. Nothing.
2. Left Attack.
3. Right Attack.
4. Left Block.
5. Right Block.
6. Left Fint.
7. Right Fint.
8. Grab.

	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint	Grab
Nothing	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Left Attack	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Right Attack	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Left Block	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Right Block	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Left Fint	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Right Fint	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Grab	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Fig.2 Initial table for the agent

Artificial intelligence changes the state in the following two cases:

1) when the player performed the action and the artificial intelligence was notified;

2) every 2 seconds.

Artificial intelligence checks the received condition (for example, the player has executed blow with the left hand), addresses to the table (Fig. 2) and chooses optimum action. Then apply the selected action, after its completion receives the result and updates the table.

In this paper, the testing of artificial intelligence was performed in several ways: manually, using a simple calculation by the processor, as well as using a processor to accelerate artificial intelligence.

An example of a table with the results of manual training is shown in Figure 3.

	Row Na	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint
1	State0	1.242522	0.992239	-3.471755	-0.971019	-0.514808	0.000000	0.000000
2	State1	0.000000	1.000000	0.000000	0.000000	2.603926	0.000000	0.000000
3	State2	-0.250000	0.000000	2.000000	3.274837	0.000000	0.000000	0.000000
4	State3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	State4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	State5	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
7	State6	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Fig. 3 Table with the results of manual training

Subsequent testing of artificial intelligence using reinforcement training was performed by processing the standard method using dry processor computing.

As a result, the agent learned to interact with the player in just 2 hours and 24 minutes.

The Intel I9 11900 (11th generation) processor, which had a new architecture for artificial intelligence – Intel Deep Learning Boost – was used for re-testing.

From the presented table, artificial intelligence chose to attack with the left hand. Due to the fact that the player rarely makes a block with the right hand, the character of the computer game is injured. As for blocking, he learned to block the player's shots after 15 shots.

The results of reinforcement training using a processor to accelerate artificial intelligence are shown in Figures 4 - 7.

	Row Na	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint
1	State0	0.142857	0.285714	0.752381	0.266667	-0.257143	-0.847619	0.333333
2	State1	0.876190	2.135180	2.122956	-0.847619	0.000000	0.790476	0.000000
3	State2	0.476191	0.323810	1.067830	0.000000	1.120466	0.400000	0.000000
4	State3	2.152290	-0.361905	0.961905	-0.742857	0.485714	-1.197333	-0.876190
5	State4	-1.242607	-0.838095	1.109596	0.047619	-1.528828	0.371429	1.861733
6	State5	1.456522	2.128169	-1.876217	1.327616	-4.974322	2.183065	5.696792
7	State6	1.089053	-1.009524	0.828571	0.000000	0.942857	0.000000	0.000000

Fig. 4. The results of experiments

	Row Na	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint
1	State0	0.209524	0.761905	0.466667	-0.114286	-0.247619	0.085714	0.647619
2	State1	0.152381	0.723810	0.428571	-0.028571	0.009524	-0.200000	-0.066667
3	State2	0.171428	0.047619	-0.047619	0.161905	-0.076190	0.380952	-0.104762
4	State3	0.114286	0.266667	0.342857	0.047619	0.152381	-0.095238	-0.019048
5	State4	0.228571	0.161905	-0.161905	0.095238	0.171429	0.047619	-0.114286
6	State5	0.000000	0.466667	0.104762	0.400000	-0.200000	0.285714	0.057143
7	State6	-0.152381	0.161905	0.238095	0.200000	0.152381	0.000000	0.000000

Fig. 5. The results of experiments

	Row Na	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint
1	State0	0.000000	0.495238	0.390476	0.333333	-0.314286	0.000000	0.000000
2	State1	0.000000	0.000000	0.228571	-0.028571	0.247619	-0.085714	0.047619
3	State2	0.000000	0.619048	0.438095	0.000000	0.238095	-0.438095	-0.180952
4	State3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	State4	0.200000	-0.114286	0.266667	-0.238095	0.285714	0.200000	0.133333
6	State5	1.131137	0.885714	0.085714	0.057143	0.400000	0.057143	0.000000
7	State6	0.000000	0.000000	0.000000	0.000000	0.514286	0.314286	0.000000

Fig. 6. The results of experiments

	Row Na	Nothing	Left Attack	Right Attack	Left Block	Right Block	Left Fint	Right Fint
1	State0	0.961905	-0.171429	0.314286	-0.495238	-0.276190	0.238095	0.209524
2	State1	0.266667	0.257143	0.219048	-0.028571	-0.190476	-0.085714	0.180952
3	State2	0.095238	0.238095	0.133333	0.200000	0.171429	-0.152381	-0.161905
4	State3	0.857143	-0.523810	0.466667	0.400000	-0.400000	0.609524	-0.295238
5	State4	0.323810	-0.257143	0.000000	0.342857	-0.514286	0.523810	0.495238
6	State5	-0.171429	0.104762	0.200000	0.152381	0.285714	0.257143	-0.295238
7	State6	-0.161905	-0.314286	0.133333	0.161905	0.209524	0.114286	0.238095

Fig. 7. The results of experiments

After analyzing the results of artificial intelligence training using all three methods, we can conclude that the use of Intel I9 11900 processor (11th generation), which had a new architecture for artificial intelligence - Intel Deep Learning Boost was the most effective.

Retraining using this hardware solution took only 1 hour and 7 minutes, and the released resources could be reallocated to other processes.

The effectiveness of the proposed solutions has been proven by experiments. The use of the model of the artificial intelligence accelerator allowed to accelerate the learning of the character of the computer game by 2.14 times compared to the classical methods.

Conclusions

A review of the literature has shown that today, given the complexity of computational processes and the high cost of these processes, the gaming computer industry needs to improve hardware and software to increase the efficiency and speed of processing artificial intelligence algorithms. An analysis of existing machine learning tools and existing hardware solutions to accelerate artificial intelligence. A reasonable choice of hardware solutions that are most effective for the implementation of the task. Possibilities of practical use of the artificial intelligence accelerator are investigated.

The effectiveness of the proposed solutions has been proven by experiments. The use of an artificial intelligence accelerator model allowed to accelerate the learning of a computer game character by 2.14 times compared to classical methods.

REFERENCES

1. Sze V, Chen Y-H, Yang T-J, Emer J S. Efficient processing of deep neural networks: a tutorial and survey. Proc IEEE, 2017. Vol. 105(12). Pp. 2295–2329.

2. Yao X, Zhou J, Zhang J, Boer C R. From intelligent manufacturing to smart manufacturing for industry 4.0 driven by next generation artificial intelligence and further on. In: 5th International Conference on Enterprise Systems (ES), 2017. Pp. 311 – 318.

3. Bishnoi L, Narayan Singh S. Artificial intelligence techniques used in medical sciences: a review. In: 8th International Conference on Cloud Computing, Data Science and Engineering (Confluence), 2018. Pp. 106–113.

4. Rao Q, Frtunikj J. Deep learning for self-driving cars. In: Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems—SEFAIS ’18, 2018. Pp. 35–38. doi:10.1145/3194085.3194087.

5. Baji T. Evolution of the GPU device widely used in AI and massive parallel processing. In: IEEE 2nd Electron Devices Technology and Manufacturing Conference (EDTM), 2018. Pp.7–9. doi:10.1109/EDTM.2018.8421507.

6. Shawahna A, Sait SM, El-Maleh A. FPGA-based accelerators of deep learning networks for learning and classification: a review. IEEE Access, 2019. Vol. 7. Pp. 7823–7859.

7. Mittal S. A survey of FPGA-based accelerators for convolutional neural networks. Neural Comput Appl, 2018. Vol. 32(4). Pp.1109–1139.

8. Apple : (документація) / Classifying Images with Vision and Core ML – 2020. URL: https://developer.apple.com/documentation/vision/classifying_images_with_vision_and_core_ml (дата звернення: 11.02.2021).

9. Jawandhiya P. Hardware design for machine learning. Int J Artif Intell Appl (IJAIA), 2018. Vol. 9(1). Pp. 63–84.

10. Wang T, Wang C, Zhou X, Chen H. A survey of FPGA based deep learning accelerators: challenges and opportunities, CoRR, 2018. Vol. 1901.04988.

11. Rigos S. A hardware acceleration unit for face detection. In: Mediterranean Conference on Embedded Computing (MECO), Bar, 2012. Pp. 17–21.

12. Nurvitadhi E, Venkatesh G, Sim J, Marr D, Huang R, Ong Gee Hock J, Liew YT, Srivatsan K, Moss D, Subhaschandra S, Boudoukh G. Can FPGAs beat GPUs in accelerating next-gen- eration deep neural networks? In: Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays—FPGA ’17, 2017. Pp. 5–14. doi:10.1145/3020078.3021740.

13. Lacey G, Taylor G, Areibi S. Deep learning on FPGAs: past, present, and future, CoRR, 2016. Pp. 1–8. arXiv: 1602.04283.

14. Faraone J, Gambardella G, Boland D, Fraser N, Blott M, Leong PHW. Customizing low-precision deep neural networks for FPGAs. In: 28th International Conference on Field Programmable Logic and Applications (FPL), IEEE, 2018. Pp. 97–102.

15. Mogilny S.B. Machine learning using microcomputers: teaching method. way. / for ed. O.V. Lisovogo and others.

16. Cheng Kwang-Ting, Wang Yi-Chu. Using mobile GPU for general-purpose computing; a case study of face recognition on smartphones. In: Proceedings of 2011 International Symposium on VLSI Design, Automation and Test, 2011. Pp. 1–4, doi: 10.1109/VDAT.2011.5783575.

17. Ouerhani Y, Jridi M, Al Falou A. Fast face recognition approach using a graphical processing unit “GPU”. In: IEEE International Conference on Imaging Systems and Techniques, 2010. Pp.80–84.

18. Li E, Wang B, Yang L, Peng Y, Du Y, Zhang Y, Chiu Y-J. GPU and CPU cooperative accel- eration for face detection on modern processors. Presented at the 2012 IEEE International Confer- ence on Multimedia and Expo (ICME), 2012. Pp. 769–775.

19. Shah A. A, Zaidi Z. A, Chowdhry B. S, Daudpoto J. Real time face detection/monitor using raspberry pi and MATLAB. In: IEEE 10th International Conference on Application of Information and Communication Technologies (AICT), 2016. Pp. 1–4.

20. Oro D, Fernandez C, Saeta J. R, Martorell X, Hernando J. Real-time GPU-based face detec- tion in HD video sequences. In: IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011. Pp. 530–537.

21. Misra J, Saha I. Artificial neural networks in hardware: a survey of two decades of progress. Neurocomputing, 2010. Vol. 74(1–3). Pp.239–255.

22. Guo K, Zeng S, Yu J, Wang Y, Yang H. A survey of FPGA-based neural network inference accelerators. ACM Trans Reconfig Technol Syst, 2019. Vol. 12(1). Pp. 1–26.

23. Talib, M.A., Majzoub, S., Nasir, Q. et al. A systematic literature review on hardware implementation of artificial intelligence algorithms. J Supercomput, 2021. Vol. 77. Pp. 1897–1938. doi:10.1007/s11227-020-03325-8.

Yelyzaveta Hnatchuk **Єлизавета Гнатчук**	PhD, Associate Professor of Computer Engineering & System Programming Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: liza_veta@ukr.net. orcid.org/ 0000-0003-2989-3183, Scopus Author ID: 57211621395, ResearcherID: I-1504-2018, https://scholar.google.com.ua/citations?hl=ru&user=5tG01jkAAAAJ.	кандидат технічних наук, доцент кафедри комп'ютерної інженерії та системного програмування, Хмельницький національний університет, Хмельницький, Україна.
Yevheniy Sierhieiev **Євгеній Сєргєєв**	master of Computer Engineering, Computer Engineering & System Programming Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: ysierhieiev@gmail.com	магістр спеціальності комп'ютерна інженерія, Хмельницький національний університет, Хмельницький, Україна.
Alina Hnatchuk **Аліна Гнатчук**	student of Computer Engineering, Computer Engineering & System Programming Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: alinasamsungj5gold2001@gmail.com, orcid.org/0000-0003-0155-9255.	студентка спеціальності комп'ютерна інженерія, Хмельницький національний університет, Хмельницький, Україна.