


Research of the Characteristics of a Convolutional Neural Network on the ESP32-CAM Microcontroller

R. D. Sharuiiev^f,  [0009-0007-9644-6865](https://orcid.org/0009-0007-9644-6865)

P. V. Popovych^s, PhD Assoc.Prof.,  [0000-0002-1572-3127](https://orcid.org/0000-0002-1572-3127)

National technical university of Ukraine "Igor Sikorsky Kyiv polytechnic institute"  [00syn5v21](https://www.researchgate.net/profile/R-D-Sharuiiev)
Kyiv, Ukraine

Abstract—The paper is devoted to solving the problem of using neural networks for real-time image recognition on low-power portable devices running on microcontrollers. The ESP-32[®] CAM microcontroller was used as the target device, on which an artificial neural network was deployed, written using the Python[®] programming language and the Tensorflow[®] library for building neural networks. The performance of the microcontroller and personal computer for object detection using a neural network and their classification were compared in the paper. The image recognition time and percentage of correctly classified objects were compared. The paper shows that the number of training epochs affects the accuracy of object classification in the image. The obtained results show that increasing the number of training epochs increases the accuracy of object recognition using the studied neural network, but a significant increase in the number of epochs does not significantly improve recognition accuracy. The difference in the obtained results for the microcontroller and personal computer image recognition accuracy ranges from 5%.

Keywords — microcontroller; neural network; epoch; training; classification.

I. INTRODUCTION

Neural networks have become an integral part of people's daily lives. Examples of this include internet search, personal assistants, machine translation of content, smart home systems, and more [1]. Let's consider the example of smart home systems. In these systems, neural networks are used to control various subsystems, such as lighting, temperature, etc., through voice assistants or cameras and motion sensors [2]. These subsystems are called nodes [3]. Nodes, in turn, are built using various sensors, hardware, and software solutions, but they all have something in common - microcontrollers inside them. These microcontrollers are responsible for collecting, processing, and transmitting information to the main control device. It should be noted that these microcontrollers are specifically designed for use in smart home systems. Unfortunately, information about microcontroller models is not publicly available due to certain commercial secrecy.

Despite the advancement of neural networks in smart home systems, there are still unresolved issues in this direction. One of these problems is related to surveillance cameras, which have settings that allow notifications to be received when an unknown person is detected in the image. On the other hand, some cameras

may ignore certain areas of house or yard that have regular movement. Some systems use artificial intelligence to determine the location and focus on the movement so that they can send the user a video recording of the incident. In the future, much can be expected from software development. Changing the settings of a camera takes a few seconds but can have a significant impact on its performance [4]. This problem has two main directions: camera placement and object recognition.

Object recognition is a branch of artificial intelligence theory that studies methods for classifying objects [5].

Neural networks, including image classifiers, can be run on various devices that have certain advantages and disadvantages:

1. Central Processing Unit (CPU) – the basic device in a computer that performs basic logical and arithmetic operations, and has multiple cores and threads. Neural networks can work on the CPU, but their performance on the CPU can be slow due to the large number of simultaneous operations that need to be performed.
2. Graphics Processing Unit (GPU) – devices designed for processing large amounts of graphical data. Unlike CPUs, they have more cores and a parallel architecture that allows for performing



many operations simultaneously. Because of this, GPUs are an excellent choice for working with neural networks, as they accelerate training and data processing.

3. Tensor Processing Unit (TPU) is a specialized type of hardware developed by Google specifically for working with neural networks. TPUs are optimized for performing operations with tensors, which are the main components of neural networks. They provide significantly higher performance compared to CPUs and GPUs when processing large amounts of data [6].

According to the specifics of different devices, all industrial and commercial neural networks operate in real-time mode on GPUs or TPUs. There are two simple explanations for this: stability under load and speed of operation. Indeed, if we are talking about a hypothetical website that runs neural network services (such as chatbots), the neural network must handle a large number of concurrent user requests on the website and do it quickly. The task of image processing and classification is more complex and requires a more complex neural network architecture due to the absence of pre-prepared answers. Accordingly, these neural networks require specialized hardware such as server installations with TPUs as the main engine for neural network operation.

But what happens with personal devices if they need to run a neural network? To solve this problem, there are specialized blocks that are integrated into microprocessors. Let's consider such a block using the example of the Apple® A12 Bionic mobile microprocessor [7] (Fig. 1). This microprocessor has a specialized "Neural Engine" block designed to work with neural networks on its power, i.e. a low-power TPU. This solution, using tensor blocks on the microprocessor crystal, simplifies the launch and use of neural networks, making it more efficient and energy-efficient. However, microprocessors are expensive and have unnecessary functionality in smart home nodes, so using microprocessors in such devices is impractical.

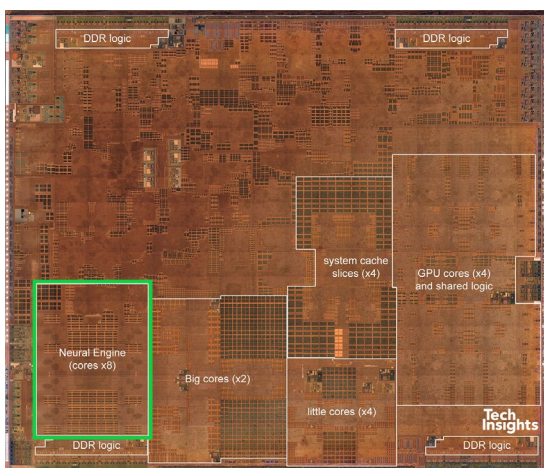


Fig. 1 The architecture of the Apple A12 Bionic microprocessor.

The research on using neural networks in autonomous compact devices is not new. One of these papers [8] focuses on the issue of drone autopiloting using artificial neural networks, but it does not describe the interaction principle between the neural network and the microcontroller or microprocessor. Another publication on this topic is a book [9] that describes the principles of applying neural networks to specialized microcontrollers in the Arduino series.

From these publications, it is evident that the problem of applying neural networks to regular microcontrollers in the consumer segment remains relevant.

To resolve the problem of deploying neural networks in smart home nodes, two approaches can be used:

1. placing separate microcontrollers for the neural network and the basic firmware is a reasonable approach, but it increases the node size, complicates its schematics, and if the device needs to be autonomous, it reduces the possible operating time from the power supply elements;
2. placing the base firmware and neural network on a single microcontroller is also a rational approach, but it requires developing a new microcontroller architecture, which is time-consuming, and costly, and the result is not always guaranteed.

Thus, there is a problem with using a neural network on a regular, non-specialized microcontroller. As an artificial neural network for image recognition in smart home video systems works based on a specialized microcontroller, developers of neural network technologies are faced with rather simple questions:

1. Can neural networks be used with ordinary microcontrollers?
2. What is the optimal level of training for a neural network to be used with microcontrollers?

This paper aims to research the possibility of using artificial neural networks with regular microcontrollers and compare the neural network's performance on a microcontroller and a personal computer.

II. EQUIPMENT AND ARTIFICIAL NETWORK DEFINITION

In this paper, a microcontroller and an artificial neural network that will run on it are used for the experiment. The artificial neural network used is a system for object recognition and classification. This system is written in the Python® [10] programming language using the Tensorflow® [11] library for building artificial neural networks.

The constructed artificial neural network has a sequential convolutional architecture with feedback connections between layers [12]. A convolutional neural

network is a class of deep artificial neural networks that utilize a type of multilayer perceptron [13] [14]. The advantage of convolutional neural networks is that they do not require a large amount of preprocessing of data before being fed into the layer of neurons, and the amount of information is constantly reduced after each layer of neurons [15].

The constructed system consists of eight layers of neurons. The cifar10 [16] dataset from the Tensorflow® library is used to train the system. This dataset contains 60000 images of size 32x32 pixels, which are divided into 10 classes (with 6000 images in each class). The selected dataset is intended for users who are just learning neural networks and therefore must meet the following criteria:

1. to have a small size;
2. to be complex (without errors or inaccuracies) and allow focusing on learning and building neural networks, rather than on the dataset itself;
3. to be embedded in the library.

To solve the first problem, the images in this dataset are pre-processed using a simplified method, including cropping the image to ensure a 1:1 aspect ratio and compressing it to 32x32 pixels. Unfortunately, the developers of this educational material did not provide information on why the image size is specifically 32x32 pixels [17].

The training set includes pre-classified images (Fig. 2). In the case of this educational material, the developers provided the following classes of objects: air-planes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

The experiment is carried out using a laptop and a microcontroller. The laptop used is the Lenovo® Legion® 5 15ACH6H [18]. A laptop, or personal computer, as a powerful computing system, is a necessary component for writing and training an artificial neural network, as well as for configuring the microcontroller and loading the artificial neural network onto it. The ESP32 CAM (Fig. 3) [19] module is used as the microcontroller. This module was chosen for its Wi-Fi module, which is necessary for downloading libraries wirelessly from the Internet, the presence of an expansion board with a programmer, and its support for the «Micropython» [20] interpreter for programming the microcontroller using the «Python» programming language. An additional advantage is the presence of the OV2640 camera [21]. The "Micro python" interpreter cannot use full-fledged neural network models and libraries for their construction. Therefore, the model training was carried out on a computer. Also, the ESP32-CAM microcontroller does not have sufficient memory capacity and computing power for training models.

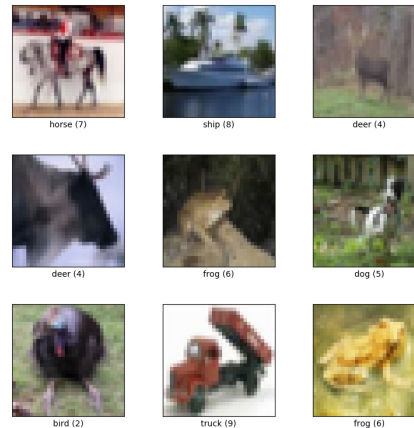


Fig. 2 Example images from the «cifar10» dataset.



Fig. 3 Appearance of the ESP32 CAM microcontroller module.

The ESP32 CAM module is designed as a training module for developing «smart home» systems, security systems, and video surveillance systems.

As an artificial neural network is a code that runs using computational resources, it is important to provide the basic characteristics of these devices in Table 1.

It should also be noted that artificial neural networks for personal computers and microcontrollers are different due to the support of different technologies. Thus, we have a different composition in the architecture of the artificial neural network. Models that run on microcontrollers and other low-power devices require some simplifications to save space that the model occupies in the memory of the microcontroller and microcontroller resources, in order not to overload its processor by 100%, and other simplifications that are automatically performed by compilers. In the case of using the Tensorflow® library, the built-in compiler turns a regular model into its lightweight version [22]. An example of simplification is shown in Fig. 4.

TABLE 1 MAIN CHARACTERISTICS OF TARGET DEVICES

| Main characteristics of the device | Device | |
|------------------------------------|--------|-----------------|
| | Laptop | Microcontroller |
| A number of cores, pcs. | 8 | 2 |
| Amount of RAM, GB | 16 | 0,00052 |
| Processor clock speed, MHz. | 3200 | 240 |



```

Перетворення моделі model_25_epochs
W tensorflow/compiler/mlir/lite/python/tf_flatbuffer_helpers.cc:364] Ignored output_format.
W tensorflow/compiler/mlir/lite/python/tf_flatbuffer_helpers.cc:367] Ignored drop_control_dependency.

```

Fig. 4 Example of recompiling a neural network to the «.tflite» format.

As a result, there are the following differences between neural networks depending on the target devices:

1. The model designed for use with a personal computer uses the «Python» interpreter and consists of five files with extensions «.pd», «.index», and «.data-00000-of-00001».
2. The model designed for use with microcontrollers and other low-power devices uses the "Micropython" interpreter, consists of one file, and has the extension ".tflite".

III. EXPERIMENT SETUP

To research the effectiveness of neural network performance when used with a microcontroller, the main characteristics for evaluation were determined, as:

1. evaluation of recognition time;
2. evaluation of the percentage of correctly classified images.

The identified characteristics are essential when working with a neural network in real-time mode. When the network is operating in real-time mode, the time required for object recognition and classification in an image depends on the processor's operating speed and the amount of RAM. The number of correctly recognized and classified objects depends on the degree of the model's learning. In turn, the degree of the model's learning depends on two main factors, assuming that all training data is valid for a particular situation:

1. the size of the training sample (the larger, the better);
2. the number of training epochs (the more, the better).

In turn, the number of training epochs carries some risks. The main risk is the overfitting of the model. To avoid this problem, the experiment is divided into 4 stages, which are necessary to determine the optimal degree of training of the network. These stages have the following distribution in Table 2.

TABLE 2 EXPERIMENT NUMBER AND NUMBER OF TRAINING EPOCHS

| Experiment number | The number of training epochs |
|-------------------|-------------------------------|
| 1 | 1 |
| 2 | 5 |
| 3 | 10 |
| 4 | 25 |

By conducting experiments with different numbers of training epochs, it is possible to determine the accuracy of the neural network's performance depending on the degree of training. Taking into account the uniformity of the neural network for the microcontroller and laptop, and the availability of data for the experiment, it is possible to determine the productivity of the neural network's work on different types of devices depending on the time spent.

The test data for verifying the neural network's performance consists of pictures of various animals and vehicles belonging to standard classes of the training material. The volume of the test sample reaches 20 pictures, with 2 pictures for each of the classes. The test sample is identical for the two target devices.

The model's runtime is measured as the total time spent on classifying the entire test set and is measured by the formula:

$$t_{sum} = \sum_{n=1}^{20} t_n,$$

where n is the index of the test sample, and t_n is the time it takes to recognize the element on the test image from the sample.

The accuracy of the model's classification is evaluated using the formula:

$$N = 100 \frac{p}{n},$$

where n is the size of the test sample and p is the number of correctly classified objects in the images.

IV. ANALYSIS OF OBTAINED RESULTS

After training the artificial neural network, data on the model's execution time and accuracy were obtained for both the computer and the microcontroller. The obtained data were recorded in Table 3.

From the obtained results, it can be seen that the execution time on the microcontroller is longer, and the accuracy is lower. Such results are expected and quite logical. The reasons for such a result are:

1. the working frequency of the microcontroller is lower, which makes the artificial neural network work slower;

TABLE 3 EXPERIMENTAL RESULTS

| Experiment number | The obtained results | | | |
|-------------------|----------------------|-------------|-----------------|-------------|
| | Laptop | | Microcontroller | |
| | Time, (s) | Accuracy, % | Time, (s) | Accuracy, % |
| 1 | 0,99 | 25,00 | 2,54 | 20,00 |
| 2 | 0,91 | 45,00 | 2,31 | 40,00 |
| 3 | 0,89 | 55,00 | 2,36 | 50,00 |
| 4 | 0,91 | 55,00 | 2,47 | 50,00 |



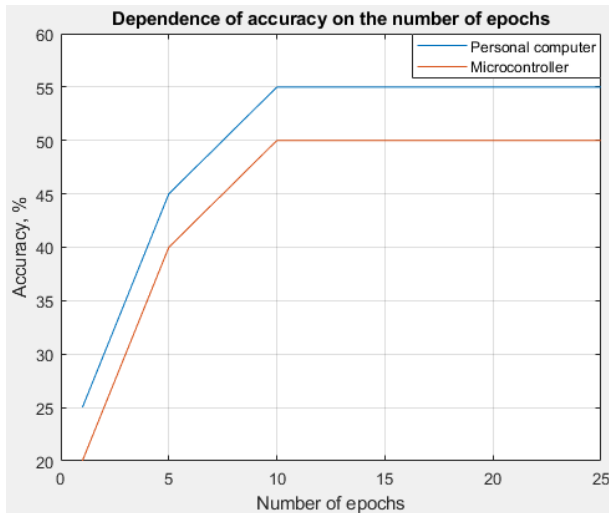


Fig. 5 Dependency graph of accuracy on the number of epochs

- the amount of RAM is smaller, so the memory gets filled with data faster, and as a result, the operating system in the form of the «Micropython» interpreter waits for the memory to be released, so during these moments, the program «freezes» due to a lack of resources;
- a smaller number of physical cores leads to a smaller number of parallel computations, which affects the device's speed;
- the lower accuracy of the work is due to the simplified architecture of the artificial neural network during conversion to a format supported by microcontrollers and other low-power devices.

It is also worth noting the lack of progress in training the artificial neural network in experiment number 4. Fig. 5 shows that the accuracy percentage of the neural network did not improve during the model's training. Such behavior can be explained by the fact that the model is overfitting, during which it began to «memorize» the answers [23]. Solutions of this problem may include:

- changing the ratio of the training and validation sets during the model's training;
- changing the number of epochs for training the model;
- attempting to train the model on a larger dataset.

The first approach is actively used in real life. In this case, the ratio of the validation set to the training set is taken from one to five, or from one to ten.

The second approach to solving this problem requires more powerful computing equipment and training

the model on graphics cards rather than central processors because during model training, floating-point numbers are used, and graphics cards perform better with these numbers, making them faster.

The third approach requires creating a custom dataset for training the neural network, which requires a large number of relevant specialists and significant time investments to find and prepare the appropriate material for model training.

During the analysis of the obtained data, the best results were achieved in experiment number 4. In this experiment, the microcontroller attained an accuracy of 50% correct answers. Speaking more precisely, the classification accuracy varied depending on the class of images: airplanes, cars, ships, trucks, and birds achieved 100% correct object classification, while for cats, deer, dogs, frogs, and horses, the classification accuracy was 0% correct answers. The data reveals a trend where images of different types of machinery and birds are recognized best, while the accuracy of recognizing other animals is zero.

These results appear due to a certain similarity among the animals. Horses, dogs, and deer are very similar to each other in terms of neural network perception due to shared distinctive features (they have four legs, fur, etc.).

The obtained results are low. They can be improved by training the model with higher-quality data over a larger number of epochs and making certain modifications to the model architecture (adding more layers with a greater number of filters). Classification accuracy can also be improved by increasing the sharpness and resolution of the images.

CONCLUSIONS

The possibility and relevance of using artificial neural networks on microcontrollers and other low-power devices were investigated in this paper. Object recognition and classification of images served as a task for the artificial network. The study was conducted by comparing the performance of the model on a personal computer and a microcontroller.

The artificial neural network obtained during the work was written in the «Python» programming language using the library for building artificial neural networks «Tensorflow®». The model was trained using the embedded dataset «cifar10». To verify the results, 20 pictures of various objects were downloaded from open access, which are included in the classification set of the neural network.

The results obtained during the experiment showed that the use of artificial neural networks with microcontrollers is possible and appropriate, but the obtained

results are different compared to the application of neural network on personal computer. The difference in the obtained data in terms of the accuracy of the model is not significant and ranges within 5%. At the same time,

the time costs for determining objects from the test dataset for use on a microcontroller are still significant and may depend on the architecture of the artificial neural network and the microcontroller model.

REFERENCES

- [1] European Parliament, "What is artificial intelligence and how is it used?," European Parliament news, 04 09 2020. [Online]. Available: <https://www.europarl.europa.eu/news/en/headlines/society/20200827STO85804/what-is-artificial-intelligence-and-how-is-it-used>. [Accessed 01 05 2023].
- [2] DATA ECONOMY MEDIA GMBH, "AI's invisible hand on daily life," Eray Eliaçık, 09 05 2022. [Online]. Available: <https://dataconomy.com/2022/05/09/artificial-intelligence-in-everyday-life/>. [Accessed 01 05 2023].
- [3] J. Gerhart, *Home Automation & Wiring*, New York: McGraw-Hill, 1999. ISBN: 978-0070246744
- [4] Startup info, "Most Common Problems with Smart Home Technology Systems," Kossi Adzo, 09 08 2022. [Online]. Available: <https://startup.info/most-common-problems-with-smart-home-technology-systems/>. [Accessed 01 05 2023].
- [5] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, Toronto: A Wiley Interscience publication John Wiley & Sons, 1973. ISBN: 978-0-471-05669-0
- [6] Termin.in.ua, "Neural network — what it is, how it works and why it is needed.," [Online]. Available: <https://termin.in.ua/nevromerezhha/#lwptoc21>. [Accessed 03 05 2023].
- [7] The Motley Fool., "3 Things You Need to Know About Apple's A12 Bionic Chip," Ashraf Eassa , 14 09 2018. [Online]. Available: <https://www.fool.com/investing/2018/09/14/3-things-you-need-to-know-about-apples-a12-bionic.aspx>. [Accessed 03 05 2023].
- [8] K. Amer, M. Samy, M. Shaker and M. ElHelw, "Deep Convolutional Neural Network-Based Autonomous Drone Navigation," [Online]. Available: <https://arxiv.org/ftp/arxiv/papers/1905/1905.01657.pdf>. [Accessed 17 05 2023].
- [9] P. Warden and D. Situnayake, *TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers*, O'Reilly Media, 2019. ISBN: 978-1492052043
- [10] Python organization, "Python official webpage," Python.org, [Online]. Available: <https://www.python.org/>. [Accessed 13 04 2023].
- [11] Tensorflow organization, "Tensorflow official webpage," Google incorporated, [Online]. Available: <https://www.tensorflow.org/?hl=ru>. [Accessed 13 04 2023].
- [12] X. Jiang, A. Hadidp, Y. Pang, E. Granger and X. Feng, *Deep Learning in Object Detection and Recognition*, Texas: Computer Science and Engineering Department, University of Texas at Arlington, 2019. ISBN: 978-9811506512
- [13] U. Kern, "Convolutional neural network (CNN)," [Online]. Available: <https://uijwalkarn.me/2016/08/11/intuitive-explanation-convnets/>. [Accessed 17 04 2023].
- [14] M. Minsky and S. A. Papert, *Perceptrons. An Introduction to Computational Geometry*, Massachusetts: The MIT Press, 1969. ISBN: 9780262630221
- [15] Zhytomyr State Technical University, "Explanation of the principle of convolution in neural networks," [Online]. Available: <https://conf.ztu.edu.ua/wp-content/uploads/2019/02/45-1.pdf>. [Accessed 19 04 2023].
- [16] Tensorflow organization, "Cifar10 dataset," Google incorporated, [Online]. Available: <https://www.tensorflow.org/datasets/catalog/cifar10?hl=ru>. [Accessed 13 04 2023].
- [17] A. Krizhevsky, V. Nair and G. Hinton, "Explanation of the CIFAR-10 dataset," [Online]. Available: <https://www.cs.toronto.edu/~kriz/cifar.html>. [Accessed 19 04 2023].
- [18] Lenovo corporation, "Lenovo Legion," Lenovo corporation, [Online]. Available: <https://www.lenovo.com/ua/uk/laptops/legion-laptops/legion-5-series/Legion-5-15ACH6H/p/88GMY501582>. [Accessed 13 04 2023].
- [19] AI-Thinker, "ESP-32 CAM datasheet," [Online]. Available: https://docs.ai-thinker.com/media/esp32/docs/esp32-cam_product_specification_zh.pdf. [Accessed 14 02 2023].
- [20] Micropython organization , "Micro Python," [Online]. Available: <https://micropython.org/download/esp32/>. [Accessed 20 02 2023].
- [21] Omnivision, "OV2640 2Mp/FOV120-NV datasheet," [Online]. Available: <https://www.ovt.com/wp-content/uploads/2021/01/OV2640-Volume-Production-FINAL.pdf>. [Accessed 20 02 2023].
- [22] Tensorflow organization, "Tensorflow lite compiler," Google incorporated, [Online]. Available: <https://www.tensorflow.org/lite?hl=en>. [Accessed 28 04 2022].
- [23] V. Chandra and A. Hareendran, *Artificial Intelligence and Machine Learning*, Delhi: PHI Learning, 2014.
- [24] Evergreen, "What are convolutional neural networks and where are they applied?," [Online]. Available: <https://evergreens.com.ua/ua/articles/cnn.html>. [Accessed 17 04 2023].
- [25] 130.com.ua, "What is an onboard computer and why is it necessary?," [Online].



Available: <https://130.com.ua/uk/what-is-trip-computers/>. [Accessed 11 04 2023].

[26] A. H. Habibi and H. E. Jahani, *Guide to Convolutional Neural Networks*, Spain: Springer International Publishing, 2017.
ISBN: 978-3319575490


Надійшла до редакції 20 квітня 2023 року
Прийнята до друку 19 травня 2023 року

Дослідження характеристик згорткової нейронної мережі на мікроконтролері ESP32-CAM

Р. Д. Шаруєв^f,  [0009-0007-9644-6865](https://orcid.org/0009-0007-9644-6865)

П. В. Попович^s, к.т.н. доц.,  [0000-0002-1572-3127](https://orcid.org/0000-0002-1572-3127)

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"  [00syn5v21](https://ror.org/00syn5v21)

Київ, Україна

Анотація—Статтю присвячено вирішенню проблеми використання нейронних мереж для розпізнавання зображень у режимі реального часу на малопотужних портативних пристроях, що працюють на мікроконтролерах. Як цільовий пристрій використано мікроконтролер ESP-32[®] CAM, на якому розгорнуто штучну нейронну мережу, написану з використанням мови програмування Python[®] і бібліотеки Tensorflow[®] для побудови нейронних мереж. У роботі виконано порівняння продуктивності мікроконтролера та персонального комп'ютера для виявлення об'єктів за допомогою нейронної мережі та їх класифікації. Порівняння проведено за часом розпізнавання зображення та відсотком правильно класифікованих об'єктів. У статті показано, що кількість епох навчання впливає на точність класифікації об'єктів на зображенні. Отримані результати демонструють, що збільшення кількості епох навчання підвищує точність розпізнавання об'єктів за допомогою досліджуваної нейронної мережі, але значне збільшення кількості епох не призводить до суттєвого покращення точності розпізнавання. Під час навчання нейромережі виявлено певне «перенавчання» моделі. Дана проблема виникла через велику кількість епох навчання на малій кількості навчальних даних для створеної моделі. Для уникнення цієї проблеми в майбутньому рекомендовано брати більше навчального матеріалу або спростити модель для використання уже існуючого набору навчальних даних. Аналіз отриманих даних показав, що відмінність у точності роботи між мікроконтролером та персональним комп'ютером склала 5%, що не є суттєвим, проте відмінність у затраченому часі склала більше ніж у 2 рази, що є критичним для класифікації об'єктів у режимі реального часу. Результати роботи моделі на мікроконтролері показали точність, яка складає 50%. Точність в залежності від класу така: для літаків, автомобілів, кораблів, вантажівок та птахів точність класифікації об'єктів склала 100%, а для тварин точність класифікації об'єктів склала 0%. З отриманих даних видно тенденцію, що найкраще розпізнаються образи техніки різних типів та птахи, а точність розпізнавання тварин є нульовою. Також наведено рекомендації з удосконалення роботи моделі, які б підвищили її точність.

Ключові слова — мікроконтролер; нейромережа; епоха; навчання; класифікація.

