Intelligent information systems

UDC 004.9 **doi:** https://doi.org/10.20998/2522-9052.2024.3.06

Elshan Hashimov[1, 2], Giblali Khaligov[2]

[1] Azerbaijan Technical University, Baku, Azerbaijan
[2] National Defense University, Baku, Azerbaijan

THE ISSUE OF TRAINING OF THE NEURAL NETWORK FOR DRONE DETECTION

Abstract. In the article, the issue of mistaking birds for UAVs, which is considered one of the main problems in the process of detecting unmanned aerial vehicles (UAVs) through cameras, was considered. **The object of the study** is a method of training an artificial intelligence model so that the detection system does not mistake drones for birds. In order to solve this problem, as an alternative to creating a new complex model, the retraining method was used by making some changes to the previously developed model. **The aim of the study** is to train an artificial intelligence model using certain methods to detect drones and prevent them from being mistaken for birds. **The main result of the research** is the training of the artificial intelligence model with the application of the proposed methods for training the model and increasing the accuracy indicators. Also, a dataset was created for both classes, an algorithm was written to retrain the pre-trained model, a fine-tuning process was performed on the model to improve the training and validation performance, and the results were graphically expressed.

Keywords: computer vision; artificial intelligence; train; dataset; deep learning.

Introduction

With the continuous development of technology, drone companies produce various types of unmanned aerial vehicles (UAV) or systems. Due to their availability and ease of use, UAVs are widely used for commercial purposes such as monitoring of public places, delivery, research, cartography, rescue and search, first aid, and agriculture [1, 2]. However, the wide and rapid spread of UAVs poses a threat when they are used in crimes such as smuggling (transporting illegal cargo at borders, restricted areas, prisons, etc.), illegal video surveillance, and interference. To ensure safety, some manufacturing companies have established no-fly zones that prohibit drones from flying within a 25 km radius of several sensitive areas such as prisons, airports, power plants, and other critical facilities. However, the effect of no-fly zones is quite limited, and not all drones have this built-in protection. Therefore, to solve this problem, anti-drone systems are intensively developed, and the problem of real-time drone detection becomes relevant [3].

Based on research, the main techniques that can be used for UAV detection and classification tasks are radar, radio frequency (RF), acoustic, and camera systems supported by computer vision algorithms [4, 5].

Radar-based drone detection. Radar is considered a traditional system that provides detection of objects flying at long distances even in unfavorable light and weather conditions. Since radar systems are mainly designed to detect high-velocity ballistic trajectory targets such as aircraft, military drones and missiles, they are not suitable for detecting small UAVs flying with relatively low-velocity non-ballistic trajectories [6]. Although radar systems are recognized as a reliable solution for detection, they have lack object classification capabilities. UAVs and birds have similiar characteristics and it is very difficult to distinguish them [7]. It makes them a less profitable solution for detection. The structural complexity and high cost of radar systems are other reasons that necessitate the creation of a relatively inexpensive drone detection system. In general, this method has the following pros and cons:

Pros:

1. Works reliably in any weather conditions.
2. Can give the exact coordinates of the object.
3. Determines the speed of movement of the object.
4. Determines the distance to the object.
5. Can detect several flying objects.
6. It covers a large area.

Cons:

1. Information about an object becomes more difficult as its speed increases.
2. Large objects perform poorly when too close to the radar.
3. It cannot provide information about the structure and type of the object.
4. When there are multiple targets, the radio signals may not distinguish the objects from each other.
5. Can be silenced by other signals.
6. It is an active system, that is, it emits radio waves from itself.

Drone detection based on acoustics. Relatively low-cost acoustic detection systems use acoustic sensors (microphones) to recognize the sound waves of UAV rotors even in poor visibility [8]. The maximum operating range of these systems is in the range of 200-250 m. On the other hand, acoustic systems are generally resistant to environmental parameters, but their limited effective range makes them less desirable. In addition, these systems are sensitive to environmental noise, especially in urban or noisy high-altitude areas and in windy conditions, affecting to the detection.

Drone detection based on radio frequency. This system is one of the most outstanding anti-drone systems. They detect and classify drones based on their radio signals. A radio-based detection system is a passive listener between the UAV and its ground controller, and unlike radar-based systems, it does not transmit any signals. Since not all drones have radio transmission, this approach is not suitable for detecting autonomous UAVs without communication.

Unlike acoustic systems, radio frequency-based systems solve the problem of limited detection range by using high-gain antennas to listen to UAV control signals, and the environmental noise problem is solved by using some noise reduction techniques [9].

Camera based drone detection. Drone detection without radio frequency transmission can be done using inexpensive camera systems based on computer vision algorithms. It is known that if target is visible, detection and recognition abilities are highest. In addition to drone detection, cameras have the advantage of providing additional information such as the drone's model, dimensions, and other visual information [10, 11]. However, detection with this method is not considered effective at night, in cloudy, foggy and dusty weather conditions. In such situations, a combination with thermal cameras can be used. Thermal cameras can solve the problem of night vision detection and sometimes even work better in rain, snow and fog, depending on the technology used. It should also be noted that high-quality thermal cameras are used for military purposes, and low-cost commercial thermal cameras may fail in high-humidity weather or other adverse environmental conditions.

The main feature of this detection process is that the distance between the camera and the object is very large. On the other hand, these objects move at a certain speed, which is not too great. For both reasons, detection becomes more difficult as the quality of the resulting images matures. The issue of object detection using computer vision has recently been reflected in many applications. These applications are used both in the household and in the industry. It should also be noted that computer vision is a branch of artificial intelligence that works through machine learning and deep learning. While artificial intelligence gives computers the ability to think, computer vision gives them the ability to see, observe and understand.

Computer vision performs object recognition in the same way as humans. A person studies the objects he sees throughout his life according to their various features and then recognizes them. However, human learning of all objects takes a lot of time. Computer vision can do this in a short period of time using cameras and algorithms. It should be noted that the main defeat of computer vision over human vision is that computer vision needs large data. Because it is enough for a person to see only one form or type of any object to be able to recognize its other forms and types. However, computer vision needs to be taught several shapes or types of that object, which increases the learning time. In object detection with computer vision, instead of its entire image, features of the images are used to recognize the object, rather than its. If those features are extracted manually, that is, by a human, this is called machine learning. In deep learning, these features are extracted by the algorithm. In both cases, those features are sent to the neural network, where the object is classified. Deep learning works with extremely large data sets and requires a large amount of computer graphics resources.

Neural networks are divided into 3 different main types depending on their purpose: Artificial neural network (ANN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). ANNs for solving complex problems, CNNs for solving problems related to computer vision, and RNNs for natural language processing are considered more suitable. A Convolutional Neural Network starts with an input images and passes it through a sequence of convolutional and pooling layers to create a feature map (Fig. 1). A convolutional layer applies several filters to the input images for extracting features. The pooling layer simplifies the feature map by retaining important information [12, 13].

A typical CNN architecture consists of three main stages: an input layer, a hidden layer, and an output layer. The input layer receives the input image and passes it to the hidden layer. The output layer defines the classification class labels and records the probability for each class. Hidden layers are an important part of CNN, the number of hidden layers and filters affects the performance of the network. A more commonly used architecture for a CNN consists of several convolutional layers, followed by one or more pooling layers, and finally a Fully connected layer to provide the output [14].

A filter is applied to input images to identify features. Known as a feature detector, this filter checks different areas of the image for a particular feature. This operation is called convolution. A filter is an array of 2D weights representing a portion of a 2D image. A filter is typically a 3×3 matrix, although there are other possible dimensions. The filter is applied to a region in the input image and calculates the dot product between the pixels. Then the filter changes position and repeats the process until it covers the entire image. The final result of all filtering processes creates a feature map.

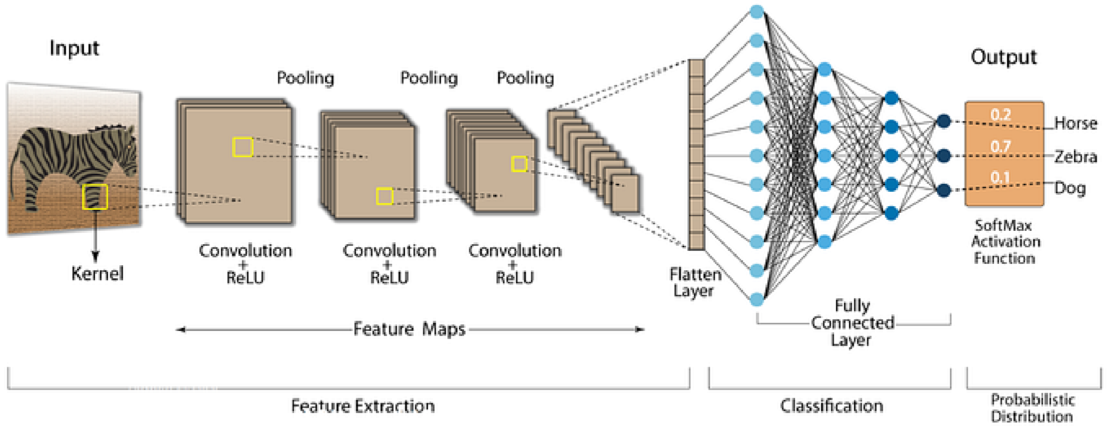

Fig. 1. The architecture of Convolutional Neural Network

A CNN typically applies a ReLU (Rectified Linear Unit) transformation to each feature map after each convolution to introduce non-linearity to the ML model. The convolution layer is usually followed by a pooling layer. Convolution and pooling layers together form a convolution block. A pooling layer reduces the size for the input of the layer that follows it. Like the convolutional operation, pooling operations use a filter to remove the entire input image, but do not use weights. There are two main types of pooling operations:

Average pooling: Calculates the average value in the matrix while scanning the filter input.

Max pooling: The filter sends the pixel with the maximum value to fill the output array. This approach is more common than average pooling.

Pooling layers are important, even though they cause some data loss, because they help to reduce the complexity and improve the efficiency of the CNN. It also reduces the risk of overfitting [15, 16].

The last layer of CNN is Fully Connected Layer. The FC layer performs classification tasks using features extracted by previous layers and filters. Instead of ReLu functions, the FC layer typically uses a softmax function that classifies the inputs more appropriately and produces a probability value between 0 and 1.

Analysis of the last research and publications

Studying the literature on this topic, analysis of articles and researches of other authors provide the article with more information. Below are some examples of the results of the literature review on drones and bird detection.

Due to the small dataset for training, the authors in [16] performed transfer learning using ZF, VGG16, and VGG M 1024 network architectures trained on the ImageNet dataset and performed evaluation. As a result, it was noted that the VGG16 architecture works more efficiently than the other two architectures. Average Accuracy scoring method was used to measure effectiveness. The evaluation shows that ZF, VGG16 and VGG M 1024 architectures are rated as 0.61(mAP), 0.66(mAP) and 0.60(mAP), respectively.

Authors Cemal Aker and Sinan Kalkan [17] created an artificial dataset by removing the background of real images, considering the lack of data for training the network. The results of the model trained by the proposed method were evaluated by the PR (precision-recall) criterion (0.9).

In [18], the authors used region-based detectors - Inception v2 and ResNet-101 architectures. Models trained with the COCO dataset were retrained with transfer learning. ResNet-101 was used due to the visibility of the drone in different sizes. The training result was 0.32 (AP) for Inception v2 architecture and 0.41 (AP) for ResNet-101.

Model training

To train the selected model, we must have a dataset of images of the object to be trained. The main issue here is the construction of a dataset consisting of images of drones and birds. At this time, it should be taken into account that if the constructed dataset has more data, the accuracy of the detector will be high. Because unlike human vision, computer vision can recognize the same object by "seeing" several different its images. However, it is not possible to increase the number of images in the dataset due to the small number of images of the object. In the paper, a dataset of 1000 images was created for both classes. For this, the method of extracting the images of objects from different video images is used. The necessary images for the dataset are collected by separating the frames from the various video images entered through the written algorithm. Also, those images were separated due to their class and stored them in a separate folder by algoritm [19].

Adaptation of the pre-trained model to a new task can be done in two stages. In the first stage (feature extraction), The representations are used which learned by a previous network to extract meaningful features from new samples. A new classifier is added, which will be trained from scratch, on top of the pretrained model so that the feature maps learned previously for the dataset can be repurposed. Do not need to (re)train the entire model. The base convolutional network already contains features that are generically useful for classifying pictures. However, the final, classification part of the pretrained model is specific to the original classification task, and subsequently specific to the set of classes on which the model was trained. The second stage (fine-tuning) consists of opening several upper layers of the frozen model base and training the newly added classification layer together with the final layers of the base model. This allows us to "fine-tune" higher-level features in the base model to better suit a specific task. In this section, the issue of training the selected MobileNet model using the generated dataset is considered. Python programming language was used for this. Before solving the problem, we need to include the libraries we will need. A new environment is created using Anaconda. We add the following libraries to that environment: opencv-python, numpy, scipy, tensorflow, keras, imutils, scikit-learn, ipython, matplotlib [20, 21]. The written algorithm is implemented in the following sequence:

1) Receiving the dataset. The dataset stored in memory is called by the algorithm fragment and the class names are added to the labels list, and the images belonging to each class are added to the data list. The images belonging to the 2 classes of the dataset are divided into two parts, one for training and another for validation. The main goal here is to avoid overfitting when the image used for training is also used for validation. Since CNN requires all the images it receives to be the same size, a new size is defined for the images. Then it is brought to the matrix form using the numpy library to perform various operations on those images [22].

2) Augmentation of received images. Since it is not considered promising to train a model with a dataset of 1000 images, methods of augmentation of those images are used. These methods include operations such as flipping, grayscale, changing saturation, changing brightness, cropping the edges with keeping the center, and rotating each image taken from the dataset. On the other hand, when training a model, it is important to consider that the object may fall into the camera lens in different shapes when detected. During detection, the

image of that object is shown to us in various forms, for example, rotated 180 degrees, mirror reflection, zooming, etc. can be given [23, 24]. Therefore, in order to increase the accuracy of training the model, augmentation methods are used in the written algorithm.

3) Loading the pre-trained base model and feature extraction. As mentioned above, the MobileNet model has been used as a pre-trained model to create the new model. This model has trained with an ImageNet dataset which has 1.4 million images and 1000 classes. First layers are choosen which is used for feature extraction. Changing the latest classification layer is not very useful. For this, the last layers before the flattened layeris used. In this layer, the features have more generality than in the upper (last) layer. Using the weights from the pre-trained ImageNet, we load the model without changing the top classification layers via the include_top=false argument:

```
base_model = tf.keras. applications.MobileNetV2(input_shape=
    IMG_SHAPE, include_top= False, weights= 'imagenet').
```

The feature extractor converts each image from 224×224×3 to a 5×5×1280 feature block. In the next step, the convolutional base is freezed and used it as a feature extractor. Additionally, a classification layer is added on top of it and trained it. With Layer.trainable = False the convolutional base is freezed so that the weights do not change during training. Sometimes some neurons are more involved in learning than others, leading to overfitting of the model, which is called overfitting. As a result, the model cannot recognize another image from the same class during testing. A Dropout layer is added to prevent overfitting during model training. The dropout layer freezes arbitrarily selected neurons during learning, enabling all neurons to participate in the learning process. In the written algorithm, 20% of the neurons in each layer are frozen.

4) Compile and train the model. Before training a model, we need to compile it. For this we use model.compile(). The learning phase is performed through model.fit(). After 10 epochs, the model is considered trained. model.save() is used to save the trained model to memory. The new model is stored in the h5 format.

5) Evaluation of the model

After building (training) the model, it is very important to evaluate it. This evaluation indicates how effectively that model works. The simplest method of evaluation is based on the number of correct classifications [25, 26].

$$accuracy = \frac{true\ predict}{number\ of\ image}.$$

For example, if 100 images from a validation dataset belonging to only one class are correctly classified in 90 cases, it can be said that the accuracy is 90%.

The graph below shows the curves of learning loss and accuracy, as well as validation loss and accuracy of the model (Fig. 2).

The initial accuracy for the trained model is 0.49, and the initial loss is 0.87.

It is an interesting fact that in the obtained graph (Fig. 2) the validation accuracy is higher than the training accuracy. This is explained by the fact that Batch Normalization and Dropout affect the accuracy during training, but they are disabled during validation.

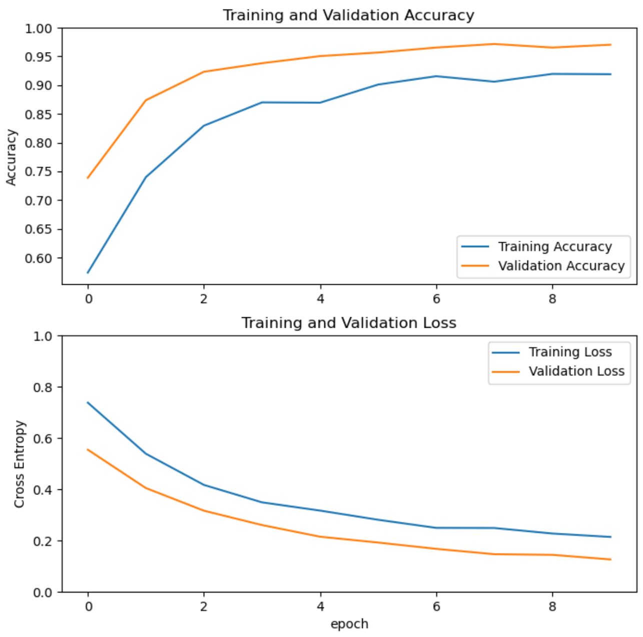

Fig. 2. Training and validation graph

The initial accuracy for the trained model is 0.49, and the initial loss is 0.87.

It is an interesting fact that in the obtained graph (Fig. 2) the validation accuracy is higher than the training accuracy. This is explained by the fact that Batch Normalization and Dropout affect the accuracy during training, but they are disabled during validation.

Through feature extraction, only a few layers of the MobileNet base model are trained. The weights of the pre-trained model remain unchanged during training. In order to improve efficiency, it is necessary to change the weights of the final layers of the base model along with training the added classification layer.

Also, try to fine-tune a small number of upper layers, not the entire MobileNet model. In most convolutional networks, the higher layer is considered the more specialized. The first few layers learn very simple and general properties that generalize almost all image types. As it goes up, the features are adapted to the dataset the model was trained on. The goal of fine-tuning is to adapt these specific features to work with the new dataset, rather than overwriting the general learning.

It needs that to do is enable base_model for training and set the bottom layers to be unchangable. Then it is necessary to compile the model again (necessary for these changes to take effect) and continue training.

The results obtained during the train are shown in Fig. 3. A vertical straight line indicates the beginning of fine-tuning.

After fine-tuning, the accuracy was increased to 0.97 and the loss was decreased to 0.04.

It is not acceptable to evaluate this model created for the detection of drones in the above-mentioned form. Because the accuracy of the model's output is fantastically high when there are no birds in the randomly taken images. On the other hand, classifying birds as drones and classifying drones as birds is not a false classification with same weight. In this evaluation, the cases of classifying drones as birds and birds as drones are not taken into account, that is, they are evaluated as the same case (false). Therefore, it is necessary to use other approaches for evaluation.

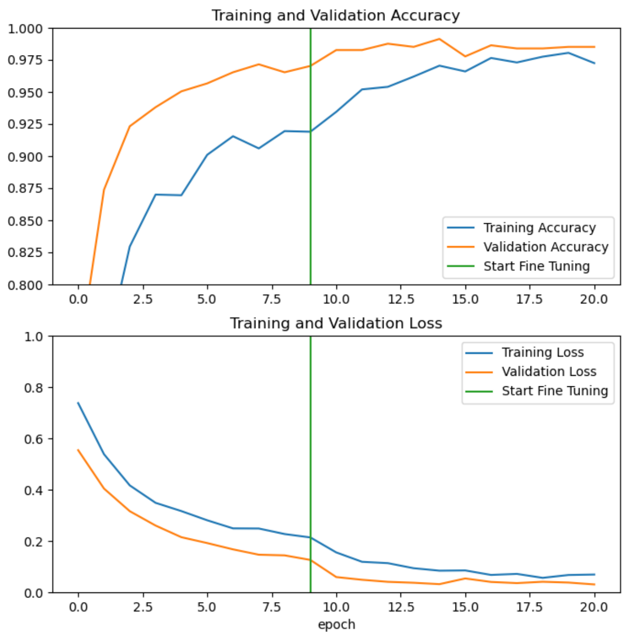

Fig. 3. Training and validation graph (fine tuning)

Taking into account the above, the confusion matrix method can be used to evaluate the model. The prediction results of 200 images used for validation are shown in the table (Table 1).

Table 1 – **Confusion matrix**

	Predicted– Drone (positive)	**Predicted– Bird (negative)**
Drone (positive)	True positives (DP) 40	False negatives (SN) 4
Bird (negative)	False positives (SP) 6	True negatives (DN) 150

$$recall = \frac{DP}{DP + SN};\ precision = \frac{DP}{DP + SP}.$$

Here, recall indicates how many drone images are classified as birds, and precision indicates how many bird images are classified as drones (recall=0.909 and precision=0.869).

In many cases, these two grades are combined into one assessment. This estimate is called $F1$.

$$F1 = \frac{2pr}{p + r}.$$

F1=0.89 for the research work.

Conclusion

The article deals with the problem of drones being mistaken for birds during the detection of drones in the airspace. It was necessary to create a model to solve the problem. Before creating the model, a dataset consisting of images of the objects to be taught by classes was created. Also, the environment necessary for writing the algorithm was created and libraries were added to that environment. The method of extracting frames from the video image was used to create the dataset. A dataset of 1000 images was created for both classes (drone and bird). Due to the fact that we have few pictures of the object to train, the method called transfer learning was used. The set goal was implemented by means of an algorithm written in Python, consisting of two parts: 1) adding a new classification layer without changing the weights, and adapting the stage of feature extraction to the new task, and 2) adapting (changing) some weights in the output layer to the new task by making fine tune. Some of the parameters from the pre-trained model were used in the new model, and some were modified for the new model.

At the end of the learning algorithm, the results for both parts are shown graphically. Finally, the accuracy of the training algorithm was 97%. Also, the model was evaluated by another evaluation method (F1=0.89).

References

1. Piriyev, H.K., Hashimov, E.G. and Bayramov, A.A. (2017), *Modelling of the battle operations*, Monograph, Herbi Nashriat, Baku, 250 p.
2. Bayramov, A.A. and Hashimov, E.G. (2018), "Assessment of invisible areas and military objects in mountainous terrain", *Defence Science Journal*, vol. 68, no. 4, pp. 343–346, doi: https://doi.org/10.14429/dsj.68.11623
3. Maharramov, R.R. and Hashimov, E.G. (2024), "Detectıon of UAVs wıth electro-optıcal system", *Current dırectıons of development of ınformatıon and communıcatıon technologıes and control tools*. Proc. of 14-th Int. Scientific and Technical Conf. ,vol. 2, pp. 74–75, doi: https://doi.org/10.32620/ICT.24.t2
4. Hashimov, E., Maharramov, R., Sabziev, E. and Pashaev, A. (2023) "Assessment of dead zone of jointly operating radars", *Control, Navigation and Communication Systems. Academic Journal*. Poltava: PNTU, 3(73), pp. 171–175, doi: https://doi.org/10.26906/SUNZ.2023.3.171
5. Hashimov, E., Sabziev, E., Huseynov, B. and Huseynov, M. (2023), "Mathematical aspects of determining the motion parameters of a target by UAV", *Advanced Information Systems*, vol.7, no. 1, pp. 18–22, doi: https://doi.org/10.20998/2522-9052.2023.1.03
6. Seidaliyeva, U., Akhmetov, D., Ilipbayeva, L. and Matson, E. T. (2020), "Real-Time and Accurate Drone Detection in a Video with a Static Background", *Sensors*, vol. 20, no. 14, 3856, doi: https://doi.org/10.3390/s20143856
7. Coluccia, A., Saqib, M., Sharma, N., Blumenstein, M., Magoulianitis, V., Ataloglou, D., Dimou, A., Zarpalas, D., Daras, P. and Craye, C. (2019), "Drone-vs-Bird Detection", *16th IEEE International Conference on Advanced Video and Signal Based Surveillance* (AVSS), Taipei, Taiwan, doi: https://doi.org/10.1109/AVSS.2019.8909876
8. Thomas, C. (2011), *Sensor Fusion – Foundation and Applications*, InTech, Rijeka, 240 p., doi: https://doi.org/10.5772/680
9. Ezuma, M., Erden, F., Anjinappa, C.K., Ozdemir, O. and Guvenc, I. (2019), "Micro-UAV Detection and Classification from RF Fingerprints Using Machine Learning Techniques", *IEEE AERO*, USA, doi: https://doi.org/10.1109/AERO.2019.8741970
10. Wu, M., Xie, W., Shi, X., Shao, P. and Shi, Z. (2018), "Real-Time Drone Detection Using Deep Learning Approach", 3rd international conference MLICOM, Hangzhou, China, pp. 22–32, doi: https://doi.org/10.1007/978-3-030-00557-3_3

11. Bayramov A. A., Hashimov, E.G. and Nasibov Y. A. (2021), "Unmanned Aerial Vehicle Applications for Military GIS Task Solutions", *Research Anthology on Reliability and Safety in Aviation Systems, Spacecraft, and Air Transport*, IGI Global, pp. 1092–1115, doi: https://doi.org/10.4018/978-1-5225-7709-6.ch010
12. Taha, B. and Shoufan, A. (2019), "Machine Learning-Based Drone Detection and Classification: State-of-the-Art in Research", *IEEE Access*, vol. 7, Toronto, Canada, pp.138669-138682, doi: https://doi.org/10.1109/ACCESS.2019.2942944
13. Ren, S., He, K., Girshick, R. and Sun, J. (2016), "Faster R-CNN. towards real-time object detection with region proposal networks", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, is. 6, China, pp. 1137–1149, doi: https://doi.org/10.1109/TPAMI.2016.2577031
14. Park, J., Kim, D.H., Shin, Y.S. and Lee, S. (2017), "A Comparison of Convolutional Object Detectors for Real-time Drone Tracking Using a PTZ Camera", *ICCAS*, Jeju, South Korea, doi: https://doi.org/10.23919/ICCAS.2017.8204318
15. Elgendy, M. (2020), *Deep learning for vision system*, Manning, 480 p., available at: https://www.oreilly.com/library/view/deep-learning-for/9781617296192/
16. Saqib, M., Daud Khan, S., Sharma, N. and Blumenstein, M. (2017), "A study on detecting drones using deep convolutional neural networks", *Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance* (AVSS), Lecce, Italy, 29 August–1 September 2017, pp. 1–5, doi: https://doi.org/10.1109/AVSS.2017.8078541
17. Aker, C. and Kalkan, S. (2017), "Using deep networks for drone detection", *Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance* (AVSS), Lecce, Italy, 29 August–1 September 2017; pp. 1–6, doi: https://doi.org/10.1109/AVSS.2017.8078539
18. Nalamati, M., Kapoor, A., Saqib, M., Sharma, N. and Blumenstein, M. (2019), "Drone Detection in Long-Range Surveillance Videos", *Proceedings of the 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance* (AVSS), Taipei, Taiwan, 18–21 September 2019, pp. 1–6, doi: https://doi.org/10.1109/AVSS.2019.8909830
19. Davies, E.R. (2012), *Computer and machine vision: Theory, Algorithms, Practicalities*, Academic Press, USA, doi: https://doi.org/10.1016/C2010-0-66926-4
20. Antonio, G. and Sujit, P. (2017), *Deep Learning with Keras*, Birmingham, UK, ISBN:978-1-78712-842-2, 318 p., available at: https://dl.acm.org/doi/10.5555/3153803
21. (2018), *Tensorflow Tutorial*, pp. 3–7, available at: https://www.tutorialspoint.com/tensorflow/index.htm
22. Chollet, F. (2021), *Deep Learning with Python*, Second Edition, 504 p., ISBN number 9781617296864, available at: https://www.manning.com/books/deep-learning-with-python-second-edition
23. Chen, Y., Aggarwal, P., Choi, J. and Kuo, C.-C. J. (2017), "A Deep Learning Approach to Drone Monitoring", *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference* (APSIPA ASC), IEEE, Kuala Lumpur, Malaysia, doi: https://doi.org/10.1109/APSIPA.2017.8282120
24. Kuchuk, H., Kovalenko, A., Ibrahim, B.F. and Ruban, I. (2019), "Adaptive compression method for video information", *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 8(1), pp. 66–69, doi: http://dx.doi.org/10.30534/ijatcse/2019/1181.22019
25. Dotsenko, N., Chumachenko, I., Galkin, A., Kuchuk, H. and Chumachenko, D. (2023), "Modeling the Transformation of Configuration Management Processes in a Multi-Project Environment", *Sustainability (Switzerland)*, мol. 15(19), 14308, doi: https://doi.org/10.3390/su151914308
26. Sokolova, M., Japkowicz, N. and Szpakowicz S. (2016), "Beyond Accuracy, F-score and ROC: a Family of Discriminant Measures for Performance Evaluation", *Advances in Artificial Intelligence*, vol. 4304, pp. 1015–1021, doi: https://doi.org/10.1007/11941439_114

Received (Надійшла) 15.04.2024
Accepted for publication (Прийнята до друку) 31.07.2024

Відомості про авторів/ About the Authors

Гашимов Ельшан Гіяс – доктор національної безпеки та військових наук, професор, професор (науковий секретар), Національний університет оборони, професор Азербайджанського технічного університету, Баку, Азербайджан;
Elshan Hashimov – Doctor in national security and military sciences, professor, Academic secretary of National Defense University, professor of Azerbaijan Technical University, Baku, Azerbaijan;
e-mail: hasimovel@gmail.com; ORCID Author ID: http://orcid.org/0000-0001-8783-1277;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57195631270.

Халігов Гіблалі – ад'юнкт, Національний університет оборони, Баку, Азербайджан;
Giblali Khaligov – Phd student, National Defense University, Baku, Azerbaijan;
e-mail: qibleli94@gmail.com; ORCID Author ID: http://orcid.org/0009-0004-1528-7822.

Навчання нейронної мережі для виявлення дронів

Е. Г. Гашимов, Г. Халігов

Анотація. У статті було розглянуто питання прийняття птахів за БПЛА, яке вважається однією з головних проблем у процесі виявлення безпілотних літальних апаратів (БПЛА) за допомогою камер. **Об'єктом дослідження** є метод навчання моделі штучного інтелекту таким чином, щоб система виявлення не сприймала дрони за птахів. Для вирішення цієї проблеми, як альтернатива створенню нової комплексної моделі, був застосований метод перенавчання шляхом внесення деяких змін у раніше розроблену модель. **Мета дослідження** полягає в тому, щоб навчити модель штучного інтелекту за допомогою певних методів виявляти дрони та запобігати тому, щоб їх сплутали з птахами. **Основним результатом дослідження** є навчання моделі штучного інтелекту із застосуванням запропонованих методів навчання моделі та підвищення показників точності. Крім того, було створено набір даних для обох класів, написаний алгоритм для повторного навчання попередньо навченої моделі, виконано процес тонкого налаштування моделі для покращення ефективності навчання та перевірки, а результати було виражено графічно.

Ключові слова: комп'ютерний зір; штучний інтелект; поїзд; набір даних; глибоке навчання.