INFORMATION TECHNOLOGY

UDC 004.932
DOI: 10.15587/1729-4061.2021.243094

IMPROVEMENT OF THE MODEL OF OBJECT RECOGNITION IN AERO PHOTOGRAPHS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

Vadym Slyusar
Doctor of Technical Sciences, Professor
Research Institute Group*
Mykhailo Protsenko
PhD, Senior Researcher
Office of Special Forces*
Anton Chernukha
Corresponding author
PhD**
E-mail: an_cher@nuczu.edu.ua
Pavlo Kovalov
PhD, Associate Professor**
Pavlo Borodych
PhD, Associate Professor**
Serhii Shevchenko
PhD
Department of Fire Tactics and Rescue Operations***
Oleksandr Chernikov
Doctor of Technical Sciences, Professor
Department of Engineering and Computer Graphics
Kharkiv National Automobile and Highway University
Yaroslava Mudrogo str., 25, Kharkiv, Ukraine, 61002
Serhii Vazhynskyi
PhD, Associate Professor
Scientific Center Air Force
Ivan Kozhedub Kharkiv National Air Force University
Sumska str., 77/79, Kharkiv, Ukraine, 61023
Oleg Bogatov
PhD, Associate Professor
Department of Metrology and Life Safety
Kharkiv National Automobile and Highway University
Yaroslava Mudrogo str., 25, Kharkiv, Ukraine, 61002
Kirill Khrustalev
PhD, Associate Professor
Department of Computer-Integrated Technologies, Automation and Mechatronics
Kharkiv National University of Radio Electronics
Nauky ave., 14, Kharkiv, Ukraine, 61166
*Central Scientific Research Institute of the Army of the Armed Forces of Ukraine
Povitroflotsky ave., 28, Kyiv, Ukraine, 03049
Department of Fire and Rescue Training*
***National University of Civil Defence of Ukraine
Chernyshevska str., 94, Kharkiv, Ukraine, 61023

Detection and recognition of objects in images is the main problem to be solved by computer vision systems. As part of solving this problem, the model of object recognition in aerial photographs taken from unmanned aerial vehicles has been improved. A study of object recognition in aerial photographs using deep convolutional neural networks has been carried out. Analysis of possible implementations showed that the AlexNet 2012 model (Canada) trained on the ImageNet image set (China) is most suitable for this problem solution. This model was used as a basic one. The object recognition error for this model with the use of the ImageNet test set of images amounted to 15 %. To solve the problem of improving the effectiveness of object recognition in aerial photographs for 10 classes of images, the final fully connected layer was modified by rejection from 1,000 to 10 neurons and additional two-stage training of the resulting model. Additional training was carried out with a set of images prepared from aerial photographs at stage 1 and with a set of VisDrone 2021 (China) images at stage 2. Optimal training parameters were selected: speed (step) (0.0001), number of epochs (100). As a result, a new model under the proposed name of AlexVisDrone was obtained.

The effectiveness of the proposed model was checked with a test set of 100 images for each class (the total number of classes was 10). Accuracy and sensitivity were chosen as the main indicators of the model effectiveness. As a result, an increase in recognition accuracy from 7 % (for images from aerial photographs) to 9 % (for the VisDrone 2021 set) was obtained which has indicated that the choice of neural network architecture and training parameters was correct. The use of the proposed model makes it possible to automate the process of object recognition in aerial photographs.

In the future, it is advisable to use this model at ground stations of unmanned aerial vehicle complex control when processing aerial photographs taken from unmanned aerial vehicles, in robotic systems, in video surveillance complexes and when designing unmanned vehicle systems

Keywords: object recognition, deep convolutional neural network, aerial photograph, unmanned aerial vehicle

Received date 16.08.2021
Accepted date 20.10.2021
Published date 29.10.2021

***How to Cite**: Slyusar, V., Protsenko, M., Chernukha, A., Kovalov, P., Borodych, P., Shevchenko, S., Chernikov, O., Vazhynskyi, S., Bogatov, O., Khrustalev, K. (2021). Improvement of the object recognition model on aerophotos using deep convolutional neural network. Eastern-European Journal of Enterprise Technologies, 5 (2 (113)), 6-21. doi: https://doi.org/10.15587/1729-4061.2021.243094*

1. Introduction

The new trend in the development of machine vision and artificial intelligence consists in working out a technology of automated monitoring of critical infrastructure objects using unmanned aerial vehicles (UAVs) [1]. Such objects include enterprises of strategic importance for the economy and security of the state [2], facilities of power

engineering [3], and chemically hazardous production [4, 5]. Disruption of the normal functioning of these facilities can threaten vital national interests [6–9]. In this regard, it is necessary to develop artificial intelligence systems and improve methods of their implementation in unmanned aerial vehicle complexes (UAVCs) [10].

All types of UAVs are equipped with means of recording aerial photographs and video information. To recognize objects in aerial photographs taken with cameras installed on UAVs, it was proposed to use deep convolutional neural networks (DCNN). Therefore, the improvement of the model of object recognition in aerial photographs using the DCNNs is an urgent problem.

2. Literature review and problem statement

A pedestrian recognition model based on convolutional neural networks (CNN). was developed in [11]. It was shown that recognition and detection of pedestrians is one of the tasks of video surveillance systems, provision of automobile safety, and robotics. The variety of factors that influence pedestrian images (figure, clothing, height, lighting, background) complicates this task. However, the problem of automating the process of object recognition in aerial photographs taken from UAVs remained unresolved.

A model of choosing an adaptive detection area to ensure reliable tracking of objects was proposed in [12]. The detection process is considered there as a linear regression problem based on correlation filters. The model is trained online to distinguish the object from the background. An improved structure of two-stream recognition based on correlation filters is used. Unlike conventional methods, the object is recognized in this model using random samples or sliding windows. The target is re-detected using the adaptive window. The proposed model was tested for sensitivity in the case of noisy images. The disadvantage of this model: high computational complexity and lack of adaptability to recognition of objects in aerial photographs taken from the UAVs.

Issues of searching for the DCNN architecture are considered in [13]. This study proposes a random topology and a multiscale comparison method for recognizing the images obtained during Earth remote sensing. The proposed approach provides the best compromise between the floating-point and precision. The results of the model assessment with the use of the Vaihingen dataset confirm its effectiveness. The disadvantage of this method: the need to adapt it to recognizing object images in aerial photographs.

Detection of ships using remote sensing of optical images is discussed in [14]. It was shown that most of the existing detectors face difficulties in hard conditions when searching for and classifying small vessels. One of the causes consists in the lack of a required number of images for the CNN training. A publicly available ship tracking dataset called ShipRSImageNet is described in this study. The lack of practical application to automating the process of object recognition in aerial photographs is a disadvantage of the approach used.

It was shown in [15] that the CNNs are widely used in applications including those for image classification. However, in the case of limitations and disadvantages of a single CNN, the image recognition accuracy can be further improved by using multiple neural networks. To take advantage of multiple neural network approaches, this study proposes to apply the weighing method. Experimental results show that combining multiple neural networks provides better overall accuracy than with a single CNN. The disadvantage of this method: great computational complexity.

Anomaly detection [16] is critically important for intelligent surveillance systems. Many approaches to detecting video anomalies using deep training techniques focus on video streams from individual cameras. These deep training methods use extremely complex data. The anomaly detection model makes decisions based on high-level functions derived from selected built-in computer vision models such as object classification models. The UCSD dataset is used for training and testing the proposed method's effectiveness. Despite this, the issues of automating the process of object recognition in aerial photographs were not considered.

It was shown in [17] that many existing methods were developed for a single camera. A new multitasking method of assessing the viewpoint using camera groups was proposed. A new dataset for tracing a viewpoint at multiple aspect angles was used. Experiments with the data set used show that the proposed method is superior to the existing ones. Inability to recognize objects in aerial photographs is its disadvantage.

It should be noted that the DCNN models have become the most efficient models for the automatic classification of vegetation cover in images obtained in remote Earth sensing [18]. The proposed model classifies up to 99 % of the UC Merced dataset volume. The disadvantage of this method: the need to adapt it to recognize the images of objects in aerial photographs.

Neural networks are widely used in practical applications for robot vision [19], however, the reliability of recognition decreases because of the influence of environmental factors. These factors include lighting, background, camera orientation, etc. To solve this problem of robot vision, three neural networks are used in this study. Experimental results show that the proposed model can improve the robot vision reliability by fusing multi-neural networks of color, shape, and texture. The disadvantage of the model: inapplicability for object recognition in aerial photographs.

Analysis of [11–18] has shown disadvantages of the known models:

– computational complexity and instability of object recognition in aerial photographs taken from UAVs;

– high requirements to the absence of noise in aerial photographs;

– lack of adapted DCNNs that solve the problem of object recognition in aerial photographs.

Models using DCNNs [13] that have shown high effectiveness in object recognition are increasingly applied to processing large arrays of satellite data. Also, the DCNN models demonstrate high effectiveness in the automatic classification of objects using a group of cameras [17] or vegetation in images obtained in remote Earth sensing [18]. All this suggests that it is expedient to conduct a study on improving the model of object recognition in aerial photographs using the DCNNs.

3. The aim and objectives of the study

The study objective consisted in improving the model of object recognition in aerial photographs using the DCNNs and choice of parameters for training these nets. All this will make it possible to automate the process of object recognition in aerial photographs.

To achieve the objective, the following tasks were set:

– study the effectiveness of recognition of object images in aerial photographs using DCNNs;

– evaluate the effectiveness of object image recognition using the proposed AlexVisDrone model.

4. The study materials and methods

Let us assume that a digital camera is installed onboard the UAV. Aerial photographs are transmitted via a communication channel to the computer at the ground control station. The digital data are stored there as a file.

The object recognition task consists in assigning one specific class (a class label) to each image S in the aerial photograph:

$$P\left[\|S\|\right]=B_i, \quad (1)$$

where P is the operator characterizing the DCNN work;

B_i is the object class i=1, ..., k, (k is the number of classes, k=10).

Networks of direct data propagation [12–15] in which attribute values (the image) of the object being classified are fed to the input and a label (the class name) or a numeric class code is formed at the output are components of the DCNNs (CNNs) architecture most often used for classification. In the proposed model, an RGB image in JPEG format with a dimensionality of 227×227×3 is fed to the neural network input and a class label is attached at the output (Table 1). Different class labels are assigned to each output of the last layer. The class label is selected proceeding from the highest output value (0.1 to 0.99).

Table 1

Class parameters

Class	Class name	Label
1	Warplane	Warplane
2	Airliner	Airliner
3	Tank	Tank
4	Submarine	Submarine
5	Ship	Ship
6	Truck	Truck
7	Car	Car
8	Bus	Bus
9	Train	Train
10	Fire-engine	Fire Engine

Assessment of the model results is important in the problem of training and evaluating the effectiveness of the neural networks (DCNNs). It will be impossible to estimate the "success" of the model or compare two different models without the introduction of effectiveness indicators. Therefore, it is important to make the right choice of metrics to estimate the fulfillment of the task posed.

The following effectiveness indicators (metrics) were selected as the main indicators characterizing the process of training and estimating the DCNN effectiveness [20]:

– accuracy (precision), that is the ratio of correctly recognized objects to the total number of presumptive and true objects [20]:

$$\text{Precision}_{val}=\sum_{t=1}^{N_{val}}\frac{N_{TP_k}}{N_{TP_k}+N_{FP_k}}\cdot 100\,\%, \quad (2)$$

where N_{TP} is the number of correctly recognized objects in the image;

N_{FP} is the number of erroneously recognized objects in the image;

N_{val} is the number of images in the test sample;

t is the current image;

– sensitivity (recall) is the ratio of correctly recognized objects to the total number of objects in the image [20]:

$$\text{Reccal}_{val}=\sum_{t=1}^{N_{val}}\frac{N_{TP_k}}{N_{TP_k}+N_{FN_k}}\cdot 100\,\%, \quad (3)$$

where N_{FN} is the number of erroneously recognized objects in the image.

The study of recognition of objects in an image was carried out using the DCNN related methods in combination with the choice of optimal training parameters.

In order to automate the process of object recognition in aerial photographs, it was proposed to use the AlexNet 2012 model as a base one.

Description of architecture and mathematical apparatus used in the AlexNet 2012 *base model.*

AlexNet 2012 model was trained with the use of ImageNet (China) dataset. Let us consider the features of using this dataset in the AlexNet 2012 model.

ImageNet dataset (China).

ImageNet is a dataset of 15 million labeled high-resolution images belonging to approximately 22,000 categories. The images were collected from the Internet and labeled by people using the Amazon Mechanical Turk crowdsourcing tool [21]. The annual ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) is held within frames of the Pascal Visual Object Challenge since 2010. ILSVRC uses the ImageNet subset of roughly 1000 images in each of 1000 categories. There are approximately 1.2 million training images, 50,000 test images, and 150,000 test images in total.

ILSVRC-2010 is a version of ILSVRC for which labels of the test set are available, so this version was used for training and testing AlexNet 2012. As is known, ImageNet consists of images of various resolutions while AlexNet 2012 model requires a constant input dimension since its authors have reduced discretization of images to a fixed resolution of 256×256 [21]. To solve this problem, ImageNet rectangular images were first rescaled so that their smaller side had a length of 256, and then the central part was cut out. As a result, a 256×256 patch was produced from it and used instead of the original image. Besides, the obtained images were processed by subtracting the average activity in the training sample from each pixel. Thus, the network was trained on (centered) values of the RGB pixels.

AlexNet 2012 *architecture.*

The architecture of AlexNet 2012 DCNN is shown in Fig. 1. AlexNet 2012 contains eight layers with weight coefficients. The first five of them are convolutional and the other three are fully connected. The output data are passed through the Softmax loss function which generates 1,000 class labels. The network maximizes the multi-line logistic regression which is equivalent to maximizing the logarithm (mean for all training cases) of the probability of cor-

rect labeling over the expected distribution. The cores of the second, fourth and fifth convolutional layers are connected with only those core maps in the previous layer that are on the same GPU. Cores of the third convolutional layer are connected with all maps of cores of the second layer. Neurons in fully connected layers are connected to all neurons in the previous layer.

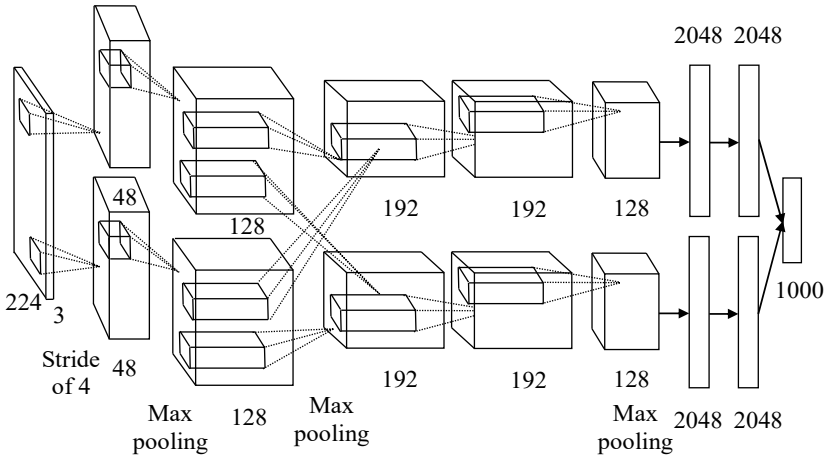


Fig. 1. Architecture of AlexNet 2012 DCNN [21]

ReLU non-linearity (function of ReLU activation).

A positively linear function *ReLU* is used in AlexNet 2012 as an activation function after each of convolutional and fully connected layers (4). The activation function *ReLU* has zero value if the argument is negative and its response is equal to the value of this argument in the case of a positive argument [21]:

$$f(s)=\max(0,s), \tag{4}$$

where $f(s)$ is the activation function; s is the argument value.

ReLU advantages:

– calculation of the sigmoid and hyperbolic tangent requires resource-intensive exponentiation operations while *ReLU* can be implemented using a simple threshold transformation of the activation matrix at zero;

– *ReLU* is not prone to saturation, it cuts off unnecessary elements in the channel at a negative input;

– the use of *ReLU* significantly increases the rate of convergence of the stochastic gradient descent (in some cases, up to 6 times) in comparison with the sigmoid and the hyperbolic tangent.

Training on several graphics processing units.

AlexNet 2012 training runs through a training set of 1.2 million images [21]. Training proceeds simultaneously on two Nvidia Geforce GTX 580 graphics processing units (GPU) because the network is split in two. This scheme reduces the error rate from 1.2 % to 1.7 % compared to a network with half the number of cores in each convolutional layer trained on a single GPU. A network with two GPUs requires slightly less training time than a single GPU network. Features of training the neural network with two processors are described in [21].

Details of AlexNet 2012 *training.*

Stochastic gradient descent (SGD) is used with a training rate of 0.01, an impulse of 0.9, and a breakup of weight coefficients of 0.0005. Weight coefficients w_{i+1} are updated according to the formula from [21]:

$$v_{i+1}=0.9\cdot v_i-0.0005\cdot c\cdot w_i-c\cdot\left\langle\left.\frac{\partial l}{\partial w}\right|_{w_i}\right\rangle_{D_i}, \tag{5}$$

$$w_{i+1}=w_i+v_{i+1}, \tag{6}$$

where i is the index of iteration;

v is an impulse variable;

c is training rate;

$\left\langle\left.\frac{\partial l}{\partial w}\right|_{w_i}\right\rangle_{D_i}$ is the weight loss gradient (average over the i-th batch D_i) of the derivative from the objective function with respect to w estimated from w_i.

Description of features of the AlexNet 2012 architecture and the mathematical apparatus used is given in [21].

Description of the Softmax function and the logistic function of loss.

Softmax activation function.

Softmax is a logistic function for the multidimensional cases used in the last layer of the DCNN. The function converts the vector s of dimensionality k into the vector f of the same dimensionality where each coordinate of the obtained vector is represented by a real number within [0,1]. The coordinate values are calculated using formula [10]:

$$f(s_i)=\frac{e^{s_i}}{\sum_{i=1}^{k}e^{s_i}}, \tag{7}$$

where $f(s_i)$ is the activation function;

i=1, ..., k is the number of classes.

The *Softmax* function is applied not to a single value but to a vector or matrix. It is used in the case of a multiclass classification problem. The network is built in such a way that the number of neurons in the last layer appears to be equal to the number of the sought classes. In this case, each neuron must produce the value of the probability of the object belonging to the class, that is, a value between zero and one, and responses of all neurons in the sum must give one.

Logistic function of losses.

The function of losses characterizes the losses caused by incorrect decision-making based on the observed data (discrepancy between the true value of the estimated parameter and the parameter estimate). To train a neural network, a logistic function is used as a loss function. It is also called a cross-entropy [23]:

$$E=-\frac{1}{N}\sum_{i-1}^{N}\left(\hat{s}_i\cdot\log(s_i)+(1-\hat{s}_i)\cdot\log(1-s_i)\right), \tag{8}$$

where s_i is the actually expected result; $\hat{s}_i$ is the model forecast.

Parameters of the activation functions used in the AlexNet 2012 architecture for the input 227×227×3 image are shown in Table 2 [23].

Table 2

Parameters of AlexNet 2012 architecture [20]

Layer (operation)	Input			Output			Core	Step, offset	Filter size		Activation function	Number of parameters
Conv 1	227	227	3	55	55	96	96	4, 0	11	11	ReLU	34,944
Max pool 1	55	55	96	27	27	96	–	2	3	3	–	–
Norm 1	27	27	96	27	27	96	–	–	–	–	–	–
Conv 2	27	27	96	27	27	256	256	1, 2	5	5	ReLU	614,656
Max pool 2	27	27	256	13	13	256	–	2	3	3	–	–
Norm 2	13	13	256	13	13	256	–	–	–	–	–	–
Conv 3	13	13	256	13	13	384	384	1, 1	3	3	ReLU	885,120
Conv 4	13	13	384	13	13	384	384	1, 1	3	3	ReLU	1,327,488
Conv 5	13	13	384	13	13	256	256	1, 1	3	3	ReLU	884,992
Max pool 3	13	13	256	6	6	256	–	2	3	3	–	–
Fc 6	9216			4096			–	–	1	1	ReLU	37,752,832
Fc 7	4096			4096			–	–	1	1	ReLU	16,781,312
Fc 8	4096			1000			–	–	1	1	Sofmax	4,097,000
Total												62,378,344

AlexNet 2012 is trained using the algorithm of error backpropagation (supervised training method) which is considered as one of the most effective DCNN training algorithms and determines the strategy of selecting weights of multilayer neural networks. Implementation features, examples of using this algorithm, and the weight setting are described in detail in [10].

Substantiation of architecture and mathematical apparatus used for implementation of the proposed DCNN.

Analysis of [20, 21] has shown that the AlexNet 2012 model has high effectiveness in the automatic classification of objects of various shapes and in various positions.

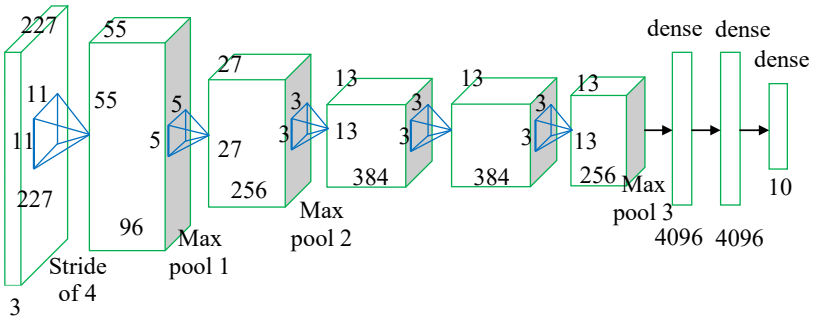


Fig. 2. Architecture of the proposed deep convolutional neural network (base AlexNet 2012)

Advantages of AlexNet 2012 and the networks based on it:

– they are invariant for various distortions, such as camera (image) rotation, varied image brightness, horizontal or vertical shifts;

– performance of AlexNet 2012 is several times higher than that of other neural networks used in the recognition problems;

– they do not require allocating much memory for storing the tags extracted during operation.

To solve the problem of object recognition in aerial photographs in 10 classes and improve the recognition effectiveness, the fully connected output layer was modified by rejecting its dimension from 1,000 to 10 neurons and additional training the modified AlexNet 2012 model in two stages. The model was additionally trained using a set of images prepared from aerial photographs at stage 1 and a set of VisDrone 2021 (China) images at stage 2. Since the model will be trained on a single GPU, the architecture will have a single branch.

The architecture of the proposed DCNN (base AlexNet 2012) is shown in Fig. 2. The problem solved by the DCNN involved the classification of objects in 10 classes.

It should be noted that the architecture of the proposed DCNN is similar to that of AlexNet 2012. The architecture parameters are shown in Table 2. The difference is the rejection of the last fully connected layer (Fc 8) up to 10 neurons and 40,960 parameters for this layer (back connections) (Table 3).

Table 3

Additional parameters of the architecture of the proposed DCNN

Layer (operation)	Number of channels		Activation function	Number of parameters (number of back connections)
	Input	Output		
Conv 1	3	96	ReLU	34,944
Max pool 1	96	96	–	0
Norm 1	96	96	–	0
Conv 2	96	256	ReLU	614,656
Max pool 2	256	256	–	0
Norm 2	256	256	–	0
Conv 3	256	384	ReLU	885,120
Conv 4	384	384	ReLU	1,327,488
Conv 5	384	256	ReLU	884,992
Max pool 3	256	256	–	0
Fc 6	9216	4096	ReLU	37,752,832
Fc 7	4096	4096	ReLU	16,781,312
Fc 8	4096	10	Sofmax	40,960
Total:				58,322,304

An RGB image of 227×227×3 dimensionality is still fed to the proposed DCNN input. The first layer convolutional filter dimensionality is 11×11. ReLU is used as the activation function and the Softmax function is used at the output (Table 3).

The third value of the input image (tensor) dimensionality (e. g. 227×227×3) is understood by some authors as the number of channels in the DCNN. In this understanding, the number of channels at the input and output of each DCNN layer is given in Table 3. As already noted, layers (operations) in the network of direct propagation follow one after another. The same pattern is valid for back connections for training, for example, there are 40,960 back connections from a fully connected layer (Fc 8) to a fully connected layer (Fc 7) and 16,781,312 back connections from a fully connected layer (Fc 7) to a fully connected layer (Fc 6) and so on.

Downsampling. Max pool.

In order to speed up the training process and reduce the consumption of computing resources, a downsampling of initial (intermediate) data is performed. There are several ways of downsampling: decimation, filtering, and pooling. The proposed model uses the maximum pooling operation (max pool). The operation of maximum pooling consists in that the so-called screening window moves along the data. The biggest pixel is chosen from the pixels falling into the window and is moved to the resulting matrix. The step (window) size in the proposed model (as well as in the base one) is equal to 2 (Table 2). Fig. 3 shows an example of the max pool operation with a 2×2 filter with a step of 2.

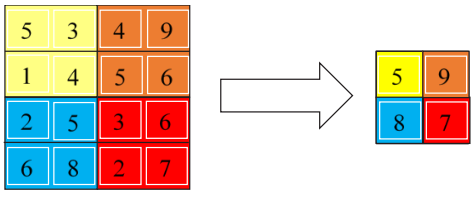


Fig. 3. Max pool operation

First convolutional layer (Conv 1).

The layer input receives an RGB image with a dimensionality of 227×227×3 which can be represented as a tensor of the third rank (valence) in the Cartesian coordinate system of 3D space. Each pixel of the 3D space in the Cartesian coordinate system has a corresponding ordered triple of real numbers, e. g. (1, 1, 1) for the first pixel and (227, 227, 3) for the last one.

The data dimensionality at the output of the convolutional layer is calculated from the formula:

$$(w,h,b)=\begin{pmatrix}(mW-kW+2\cdot p)/a+1,\\(mH-kH+2\cdot p)/a+1,b\end{pmatrix}, \quad (9)$$

where (w, h, b) is the calculated data dimensionality at the output;

mW, mH is the image width and height (of the input matrix in an individual color component);

kW, kH is the filter width and height;

p is the offset size;

a is the step size;

b is the number of cores (the number of filters).

The dimensionality of the data tensor at the layer output is calculated from formula (9). By substituting the values $mW=mH=227$, $kW=kH=227$, $a=4$, $p=0$, $b=96$ (Table 2) into formula (9), the data size (55×55×96 pixels) is obtained at the output of the first convolutional layer.

To unify the formalization of the description of neural networks of various structures and complexity, reduce the time of processing the images in the DCNN, it is proposed to analytically describe operations performed in a particular layer based on a family of penetrating end products of matrices. It has shown high effectiveness in applications of the tensor-matrix theory [23]. According to [24], the penetrating face product of the $p\times g$-matrix A and the n-dimensional tensor B wounds into a block matrix containing $p\times g$-blocks ($B=[B_n]$, $n>1$) results in a matrix of the following form:

$$A\text{¤}B=[A\circ B_n], \quad (10)$$

where $A\circ B_n$ represents the Hadamard product.

If tensor B is written as a block-row, the following is obtained:

$$A\text{¤}B=[A\circ B_r]=[A\circ B_1\ A\circ B_2\ \dots\ A\circ B_r\ \dots]. \quad (11)$$

For example:

$$A=\begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\\ a_{31} & a_{32}\end{bmatrix},$$

$$B=\left[\begin{array}{cc|cc|cc} b_{111} & b_{121} & b_{112} & b_{122} & b_{113} & b_{123}\\ b_{211} & b_{221} & b_{212} & b_{222} & b_{213} & b_{223}\\ b_{311} & b_{321} & b_{312} & b_{322} & b_{313} & b_{323}\end{array}\right],$$

$$A\text{¤}B=$$

$$=\left[\begin{array}{cc|cc|cc} a_{11}\cdot b_{111} & a_{12}\cdot b_{121} & a_{11}\cdot b_{112} & a_{12}\cdot b_{122} & a_{11}\cdot b_{113} & a_{12}\cdot b_{123}\\ a_{21}\cdot b_{211} & a_{22}\cdot b_{221} & a_{21}\cdot b_{212} & a_{22}\cdot b_{222} & a_{21}\cdot b_{213} & a_{22}\cdot b_{223}\\ a_{31}\cdot b_{311} & a_{32}\cdot b_{321} & a_{31}\cdot b_{312} & a_{32}\cdot b_{322} & a_{31}\cdot b_{313} & a_{32}\cdot b_{323}\end{array}\right]. \quad (12)$$

As applied to the first layer of the neural network under consideration, matrix A should be interpreted as a set of weight coefficients of a sliding filter used to form a convolution from the original matrix of image pixels. In this case, the matrix of an original image at the input of the neural network must be transformed into a block matrix B of pixels with the dimensionality of blocks equal to the dimensionality of the convolution filter. Pixels in neighboring blocks of the matrix are repeated with accuracy up to an offset equal to the step size when the filtering window shifts.

In addition to the matrix operation shown in (10), its generalization in the form of a direct (Kronecker) penetrating product [23] deserves attention in the context under consideration. For tensors $\boldsymbol{A}$ and $\boldsymbol{B}$ unfolded into block matrices with blocks of the same dimensionality, such version of matrix multiplication allows one to obtain the element-wise product of each block of matrix A by all blocks of the matrix B:

$$\boldsymbol{A}[\text{¤}]\boldsymbol{B}=[A_{ij}\text{¤}\boldsymbol{B}]=[A_{ij}\circ B_{mr}]. \quad (13)$$

In the proposed method of describing the neural networks, a 227×227×3 pixel image at the input is the tensor. In what follows, block matrices are used in calculations with the preservation of dimensions (by the balance of indices).

Thus, the following mathematical expression describing the procedure of processing pixels for the first convolutional layer can be written:

$$\text{Ƀ}_2=ReLU[\mathbf{1}^T[\times](\boldsymbol{A}_1[\square]\boldsymbol{B}_1)[\times]\mathbf{1}], \quad (14)$$

where $\boldsymbol{A}_1$ is the block matrix of coefficients of the first layer neurons containing 96 blocks of dimensionality 11×11;

$\boldsymbol{B}_1$ is a block matrix formed from the initial three-dimensional tensor of an RGB image analyzed by a neural network by means of slicing in it blocks of 11×11 format at a step of 4 pixels;

$\mathbf{1}^T$ and $\mathbf{1}$ are block-row and block-vector of units containing 11 unit elements in each block while the total number of blocks is consistent with the number of blocks in the column and row of the block matrix $\boldsymbol{A}_1[\square]\boldsymbol{B}_1$;

[×] is the symbol of the regular block product of matrices [23];

T is the symbol of transpose operation.

Multiplication by a block-row and a block-vector of units $\mathbf{1}^T$ and $\mathbf{1}$ actually formalizes the operation of summing the elements of the resulting matrix. As a result of the described operations, each block in the argument of the *ReLU* function in (14) turns into a scalar.

First max pooling layer (Max pool 1). A matrix of dimensionality 55×55×96 pixels is fed to the first *maxpooling* layer.

Mathematically, this layer can be described by the formula:

$$\text{Ƀ}_3=Max\ pool\ [\boldsymbol{B}_2], \quad (15)$$

where $\boldsymbol{B}_2$ is the block matrix of pixels which is formed from the block matrix of the previous layer Ƀ_2 by sliding sampling from it 27×27×96 blocks of 3×3 pixels in size with a discrete step of 2 pixels;

Max pool is the operation of selecting local maxima within each block from a number of elements greater than zero;

Ƀ_3 is the output matrix of pixels.

At the output, there is a matrix having the size of 27×27×96 pixels.

Thus, the *max pool* operation is performed similarly to the formation of a sliding convolution with overlapping of the element arrays, that is, the step size is less than the length of the pool window (the pool dimensionality is 3×3 squares and the pool step size is 2). The pool of overlap (cutting-off layer) makes it possible to avoid retraining (this property applies to all *maxpooling* layers).

Second convolutional layer (Conv 2).

The second convolutional layer receives a block matrix $\boldsymbol{B}_3$ with a dimensionality of 27×27×96 pixels and is formed from the tensor of the previous layer Ƀ_3 by sampling 5×5 blocks in it with a step of 1 pixel. Accordingly, sliding convolution with a 5×5 filter is performed further. Mathematical expression for the second convolutional layer will be as follows:

$$\text{Ƀ}_4=ReLU[\mathbf{1}^T[\times](\boldsymbol{A}_2[\square]\text{Ƀ}_3)[\times]\mathbf{1}], \quad (16)$$

where $\boldsymbol{A}_2$ is a block matrix of coefficients of the neurons of the second convolutional layer. This matrix contains 256 blocks with a dimensionality of 5×5;

$\mathbf{1}^T$ and $\mathbf{1}$ are the block-row and block-vector of units containing 5-element blocks;

Ƀ_4 is the output matrix of the second convolutional layer.

A block matrix with a dimensionality of 27×27×256 pixels is formed at the output of the second convolutional layer.

Second maxpooling layer (Max pool 2).

A matrix with a dimensionality of 27×27×256 pixels is fed to the second maxpooling layer.

This layer (*Max pool 2*) can be mathematically described by the formula:

$$\text{Ƀ}_5=Max\ pool\ [\boldsymbol{B}_4], \quad (17)$$

where $\boldsymbol{B}_4$ is a block matrix of pixels Ƀ_4 prepared for the max pool operation in which similarly to (15), 27×27×256 blocks of 3×3 pixels in size are formed with a discrete step of 2 pixels;

Ƀ_5 is the output block matrix of the layer (*Max pool 2*).

There is a matrix with a dimensionality of 13×13×256 pixels at the output of this layer.

Third convolutional layer (Conv 3).

By analogy with (14), (16), a matrix with the dimensionality of 13×13×256 pixels is fed to the third convolutional layer. Mathematical expression for the third convolutional layer can be represented as follows:

$$\text{Ƀ}_6=ReLU[\mathbf{1}^T[\times](\boldsymbol{A}_3[\square]\boldsymbol{B}_5)[\times]\mathbf{1}], \quad (18)$$

where $\boldsymbol{A}_3$ is the block matrix of coefficients of neurons of the third convolutional layer containing 384 blocks with the dimensionality of 3×3;

$\boldsymbol{B}_5$ is formed from Ƀ_5 by sampling blocks of 3×3 format with a step of 1 pixel;

$\mathbf{1}^T$ and $\mathbf{1}$ are the block-row and the block-vector of units containing 3 unit elements in each block;

Ƀ_6 is the output block matrix of the second convolutional layer.

There is a block matrix with a dimensionality of 13×13×384 pixels at the output of the third convolutional layer.

Fourth convolutional layer (Conv 4).

A block matrix with a dimensionality of 13×13×384 pixels is fed to the fourth convolutional layer. Mathematical expression for the fourth convolutional layer is similar to (18) and coincides with it in terms of dimensions of blocks in the block matrices and the principles of their formation:

$$\text{Ƀ}_7=ReLU[\mathbf{1}^T[\times](\boldsymbol{A}_4[\square]\boldsymbol{B}_6)[\times]\mathbf{1}], \quad (19)$$

where $\boldsymbol{A}_4$ is a block matrix of coefficients of neurons of the fourth convolutional layer containing 384 blocks with the dimensionality of 3×3 size;

$\boldsymbol{B}_6$ is formed from Ƀ_6 by sampling blocks of 3×3 structure with a unit step;

$\mathbf{1}^T$ and $\mathbf{1}$ are the block-row and the block-vector of units identical in structure (18);

Ƀ_7 is the output block matrix of the fourth convolutional layer with a dimensionality of 13×13×384 pixels.

The fifth convolutional layer (Conv 5).

Having the same size of the sliding filter 3×3, it differs in the number of blocks reduced to 256 in the block matrix of weight coefficients $\boldsymbol{A}_5$. Mathematical expression for the fifth convolutional layer has the following form:

$$\text{Ƀ}_8=ReLU[\mathbf{1}^T[\times](\boldsymbol{A}_5[\square]\boldsymbol{B}_7)[\times]\mathbf{1}], \quad (20)$$

where $\boldsymbol{A}_5$ is the block matrix of coefficients of neurons of the fifth convolutional layer consisting of 256 blocks with the dimensionality of 3×3;

$\boldsymbol{B}_7$ is the result of the transformation of Ƀ_7 into a set of blocks having 3×3 elements each;

$\mathbf{1}^T$ and $\mathbf{1}$ are the block row and the block vector of units in which blocks contain 3 unit elements each;

$\mathfrak{B}_8$ is the output matrix of the fifth convolutional layer.

As a result, a block matrix with a dimensionality of 13×13×256 pixels is fed to the output of the fifth convolutional layer.

The third maxpooling layer (Max pool 3)

This layer is essentially identical to the two previous Max pool layers (15), (17) with the difference that the maxpooling layer processes in this case a block matrix of data with the dimensionality of 13×13×256 elements.

The layer (*Max pool* 3) can be mathematically described by the formula:

$$\mathfrak{B}_9 = Max\ pool\ [\boldsymbol{B}_8], \quad (21)$$

where $\mathfrak{B}_9$ is the output block matrix of the layer (*Max pool* 3).

At the output, there is a block matrix with a dimensionality of 6×6×256 pixels which is converted into a vector $\boldsymbol{B}_9$ of 9,216 pixels.

First fully connected layer (*Fc*).

The first fully connected layer receives the vector $\boldsymbol{B}_9$ of 9,216 pixels which was previously transformed into a vector $\boldsymbol{B}_{10}$ of 4,096 elements using the dropout operation. The subsequent fully connected layer improves the quality of object recognition and can be described by the formula:

$$\boldsymbol{B}_{11} = ReLU[\boldsymbol{A}_6\boldsymbol{B}_{10}], \quad (22)$$

where $\boldsymbol{A}_6$ is the matrix consisting of 4096×4096 coefficients of neurons of the first fully connected layer;

$\boldsymbol{B}_{11}$ is the 4,096-pixel output vector of the first fully connected layer.

The second fully connected layer (*Fc*).

This layer functions similarly to the previous one operating with a 4,096-pixel data vector according to the formula:

$$\boldsymbol{B}_{12} = ReLU[\boldsymbol{A}_7\boldsymbol{B}_{11}], \quad (23)$$

where $\boldsymbol{A}_7$ is the matrix with the dimensionality of 4,096×4,096 with coefficients of neurons of the second fully connected layer;

$\boldsymbol{B}_{12}$ is the output vector of the second fully connected layer consisting of 4,096 pixels.

The third fully connected layer (*Fc*).

This layer is distinguished by the use of the Sofmax activation function. Its response to an input vector of 4,096 pixels has the following form:

$$\boldsymbol{B}_{13} = Sofmax[\boldsymbol{A}_8\boldsymbol{B}_{12}], \quad (24)$$

where $\boldsymbol{A}_8$ is the matrix of coefficients of neurons of the second fully connected layer with the dimensionality of 4,096×10;

$\boldsymbol{B}_{13}$ is the output vector of the third fully connected layer.

As a result, a vector of 10 values is obtained at the output. Thus, each neuron produces a value of the probability of the object belonging to a certain class, that is, a value between zero and one and all neurons in total should give a unit response.

The proposed approach makes it possible to:

– increase the DCNN speed;

– unify formalization of the description of the neural networks with different structures and complexity.

The studies and graph plotting were performed using the MATLAB R2020b computer environment for mathematical modeling (USA) and its Deep Learning Toolbox package (used as an iterator).

A model described by the authors in [10] was chosen as a neuron model. The sigmoidal (logistic) function which is the most common transfer function was used as the transfer function of neurons in the output layer. The mathematically sigmoidal (logistic) function is described in [25]. The simple form of its derivative and, accordingly, the nabla function formed on its basis is an important advantage of this function. This feature makes it easier to use the algorithm of back error propagation to train the network. In addition, due to the function shape, large pixel values grow much less than small ones. This enables the prevention of saturation from large pixel values.

To prepare images for training and testing samples, the ABBYY Screenshot Reader software (USA) was used. It makes it possible to get an image of the same dimensions and save it as a JPEG file. Preparation of the image of the "war-plane" object from the digital image of the VisDrone set is shown in Fig. 4.

Fig. 4. Preparation of an image of a "war-plane" object from a digital image of the VisDrone set using ABBYY Screenshot Reader (USA)

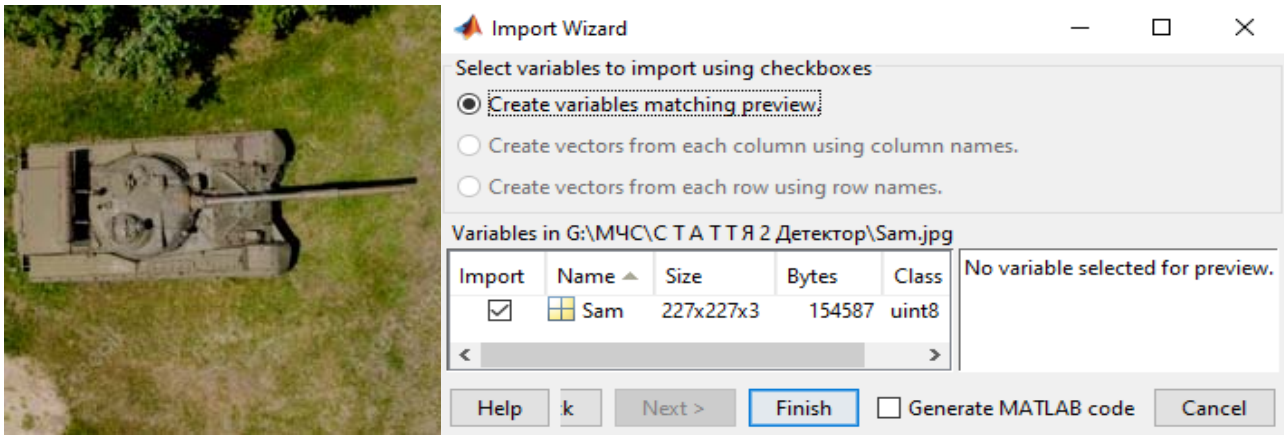
An example of importing an image from a file into the Matlab R2020b (USA) mathematical modeling environment is shown in Fig. 5.

Ways to escape retraining of the proposed AlexVisDrone model:

– improve training images. The first approach: rotate in a horizontal plane and crop (Fig. 6). The second approach: vary the brightness of the RGB image channels [21]. The third approach (recommended): combine the first and second approaches (Fig. 7);

– dropout (exclusion) is applied ahead of the first and second fully connected layers. Dropout is the DCNN regularization method designed to reduce network retraining by avoiding complex connections of individual neurons in the training data [26]. As a result, more trained neurons gain more weight in the network. This technique significantly increases the training rate, improves the quality of training with training data and the quality of the model predictions with the new test data. The technique of implementing the dropout operation for neural networks is considered in detail in [26].

Training using the images shown in Fig. 6, 7 with different angles of rotation makes it possible to adapt (train) the neural network to different orientations of the object image.



a *b*

Fig. 5. A variant of image importing: *a* — an image with the dimensionality of 227×227×3; *b* — the window of importing the image into the MATLAB R2020b mathematical modeling environment

a *b* *c* *d*

Fig. 6. Image rotation: *a* — original image position; *b* — rotated by 90°; *c* — rotated by 180°; *d* — rotated by 270°

a *b* *c* *d*

Fig. 7. Changes in the image brightness and rotation angle: *a* — original image; *b* — the image brightness increased by 5 %, rotated by 45°; *c* — the image brightness increased by 10%, rotated by 135°; *d* — the image brightness increased by 15 %, rotated by 225°

The retrained neural network was tested with ACPI X64 computer equipped with AMD Ryzen (3) 1200 Quad-Core processor with a clock frequency of 3.10 GHz and 8 GB RAM.

The studies were carried out under the following assumptions and limitations:

– a digital camera is mounted on the UAV and takes pictures in the view range;

– aerial photograph in digital form is transmitted through the communication channel to the ground control station;

– objects in the aerial photograph are recognized on the computer at the ground UAVC control station;

– the object class is recognized in the aerial photographs consecutively;

– shooting is performed in the daytime.

5. The results obtained in the study of object image recognition using DCNN

5. 1. Studying the effectiveness of recognition of the object images in aerial photographs using DCNN

To study the effectiveness of object image recognition in aerial photographs using DCNN, images of training and

test samples were prepared. 140 images for each class were used as a training sample. The number of classes totaled 10 (warplane, airliner, tank, submarine, ship, truck, car, bus, train, fire-engine). The type of training and test samples (the same): an image with the dimensionality of 227×227 in JPEG format. 70 images were used for each class as a test sample; in total, 700 images of objects were used for testing.

As already noted, the objects were recognized at the ground control station. An UAV equipped with a Sony ILCE-7M2K camera was used for shooting. This camera has the following characteristics:

– matrix type: 35 mm full-frame Exmor CMOS;

– image size (pixels): 6,000×3,376 (20Mb); 3936×2216 (8.7 Mb).

– dimensions (W×H×L): 126.9×95.7×59.7 mm;

– weight: 599 g.

An example of an aerial photograph taken with Sony ILCE-7M2K digital camera from the UAV at an altitude of 1250 meters is shown in Fig. 8.

Fig. 8. Aerial photograph taken with Sony ILCE-7M2K camera

Specifications of the digital aerial photograph:

– resolution: 6,000×3,376;

– color depth: 24 bits/pixel;

– file size: 9,665,595 Bytes;

– focal length 55 mm.

An aerial photograph can contain up to several dozen objects (Fig. 8). To prepare images of the training and test samples, the aerial photograph was scaled to such a level that the central part of the object image with a resolution of 227×227 could be cut out. ABYY Screenshot Reader (USA) (Fig. 9) made it possible to cut out an image of the same dimension 227×227×3 and save it as a JPEG file.

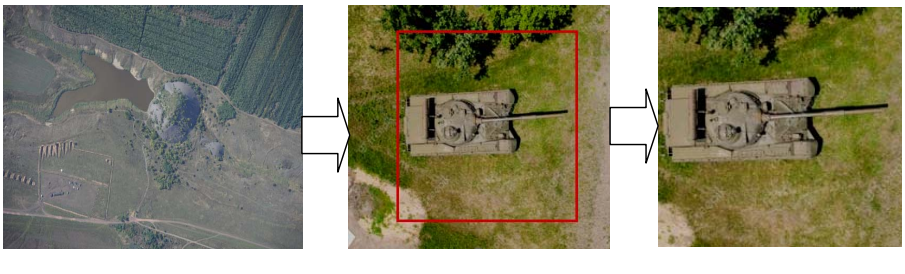


Fig. 9. Preparing an image of a "tank" object from a digital aerial photograph using ABBYY Screenshot Reader (USA)

Accuracy and sensitivity were chosen as indicators of the effectiveness of object recognition in DCNN aerial photographs. To assess the effectiveness of DCNN training, parameters of the neural network were determined. These parameters include training duration (the number of epochs), optimization algorithm (stochastic gradient descent (SGD)), training rate (training step).

The training rate (training step) is a DCNN training parameter that determines the size of the training (adjustment) step of the weight coefficients which makes it possible to control the amount of weight correction. It is selected in the range from 0 to 1. Zero is not indicated since the weight will not be corrected at all in this case. Small values of the coefficient (0.1–0.001) correspond to a smaller step of weight correction. In this case, the number of training steps increases but the accuracy of the model adjustment also increases which potentially reduces the training error.

Parameters of training the DCNN models are shown in Table 4. Three types of neural networks were compared: AlexNet 2012, VGG-S 2014 and VGG-F 2014.

Deep Training Toolbox package of the MATLAB R2020b mathematical modeling environment was used for modeling. In this package, a pretrained network can be used by selecting it from the proposed list. The network adjustment with repeated training is much faster and easier than training the network with randomly initialized weights from scratch. As a result, the network can be quickly trained for a new task using fewer images.

The study (modeling) procedure with AlexNet 2012 taken as an example.

Step 1. Download and install the Deep Learning Toolbox model for AlexNet network support package.

Step 2. Verify successful installation by entering the Alexnet command in the command line. If the required support package is installed, the SeriesNetwork function appears.

Step 3. Visualize the neural network architecture using the Deep Network Designer with the analyzeNetwork command.

Step 4. Upload new images as Datastore.

Step 5. Split the data into training and testing (validation) datasets using the splitEachLabel function.

Step 6. Boot the pre-trained network with the net=alexnet command.

Step 7. Use the analyzeNetwork command to display interactive visualization of the network architecture and detailed information about the network layers.

Step 8. Replace the last layer with the net.Layers function.

Step 9. Load the image using the imageDatastore function. The network requires input images with a dimensionality of 227×227×3 where 3 is the number of color channels. Increasing the number of images helps discourage network retraining. Divide data into training and testing datasets with the splitEachLabel function.

Table 4

Parameter combinations in the training process

Model	Training duration (number of epochs)	Algorithm of optimization	Base model	Training rate
AlexNet 2012	100	SGD	LeNet 1989	0.0001
VGG-S 2014	100	SGD	AlexNet 2012	0.001
VGG-F 2014	100	SGD	AlexNet 2014	0.001

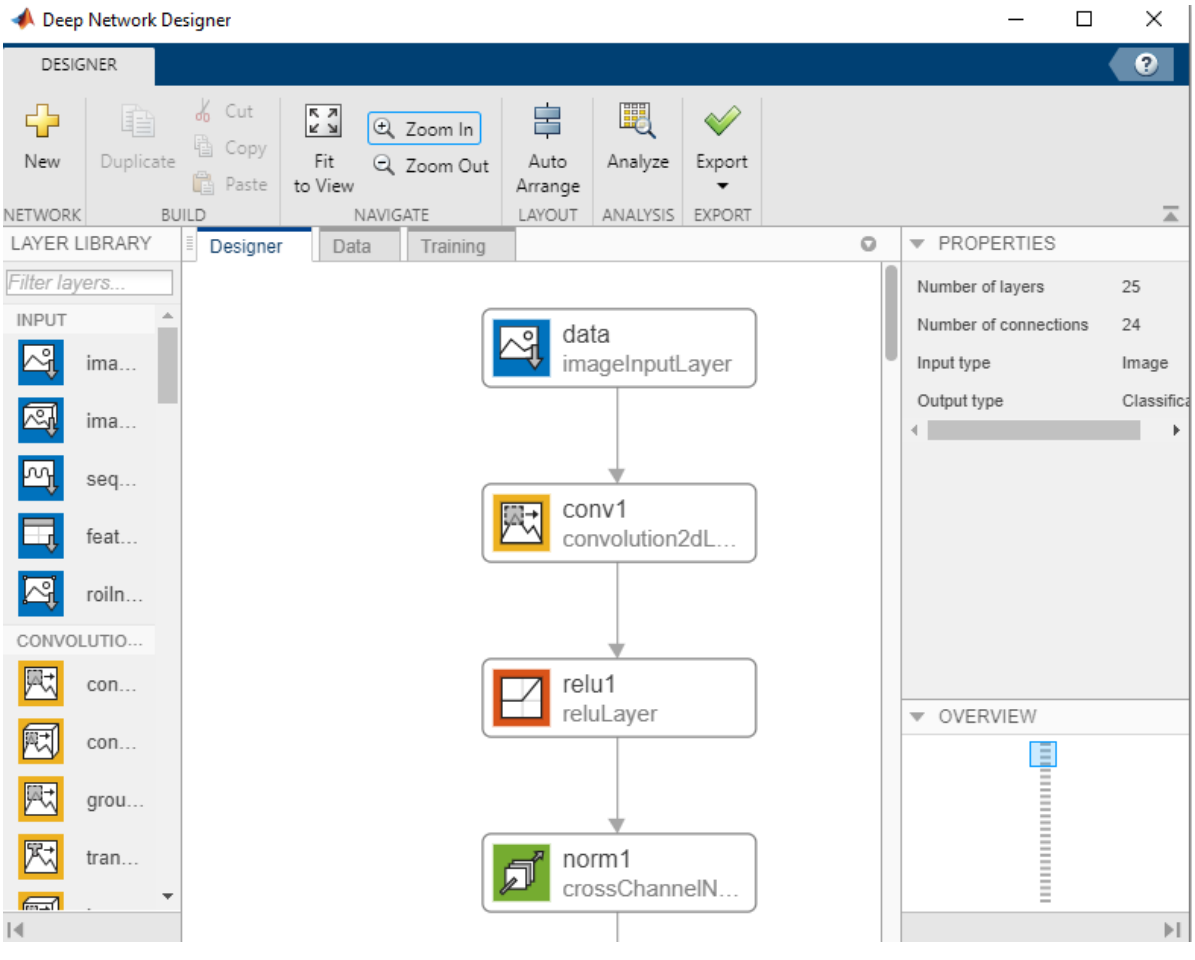


Fig. 10. Visualization of AlexNet 2012 DCNN architecture using Deep Network Designer

Step 10. Train the network by setting training parameters (options) using the trainingOptions function. If the hardware allows, trainNetwork uses the GPU by default, otherwise it uses the CPU. Training on the graphic processor requires Parallel Computing Toolbox and a supported device of the GPU.

Step 11. Classify the test sample image using the Classify function.

Step 12. Estimate the accuracy of the test sample classification using the Accuracy function.

Step 13. Estimate sensitivity of the test sample using the Recall function.

Step 14. Plot accuracy and sensitivity of the test sample using the Plot function.

If you are familiar with the DCNN architecture, steps 4, 8 can be skipped.

The studies were carried out according to the proposed procedure. Fig. 12 shows the graphs of testing the AlexNet 2012 model effectiveness.

It is seen in Fig. 12 that accuracy with the test sample stabilizes after 40 epochs and varies in the range from 84 % to 92 %. Sensitivity is 90 % for the AlexNet 2012 model and the 100th epoch.

For comparison, Fig. 13 shows graphs of checking the sensitivity of the VGG-S 2014 model.

It can be seen in Fig. 13 that accuracy with the test sample stabilizes after 30 epochs and varies in the range from 81 % to 91 %. Sensitivity for the VGG-S 2014 model begins to increase significantly after the 30th epoch and approaches 89 % after the 100th epoch.

Fig. 14 shows graphs obtained in checking effectiveness for the VGG-F 2014 model.

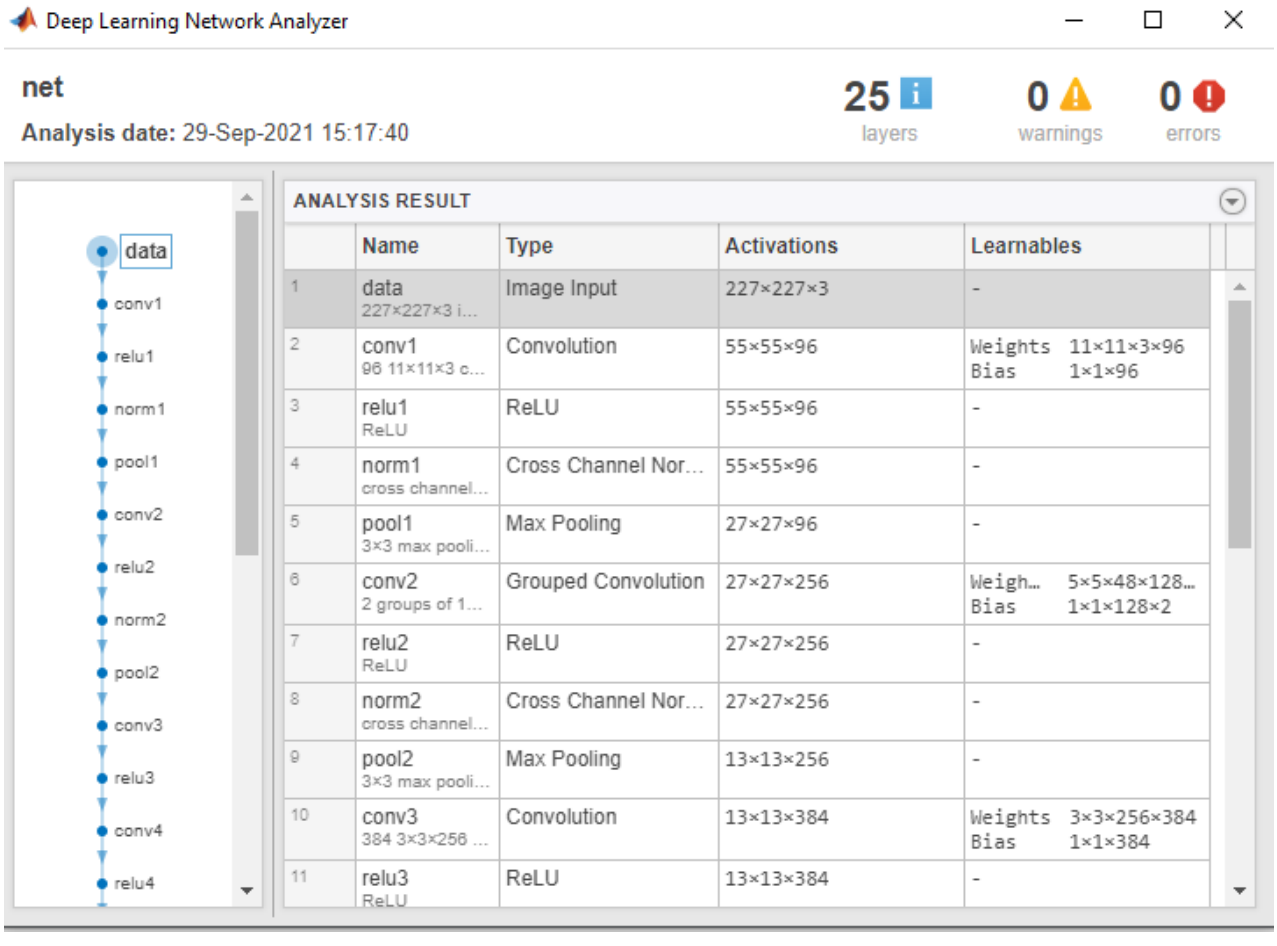


Fig. 11. Visualization of information about AlexNet 2012 network layers using Deep Training Network Analyzer

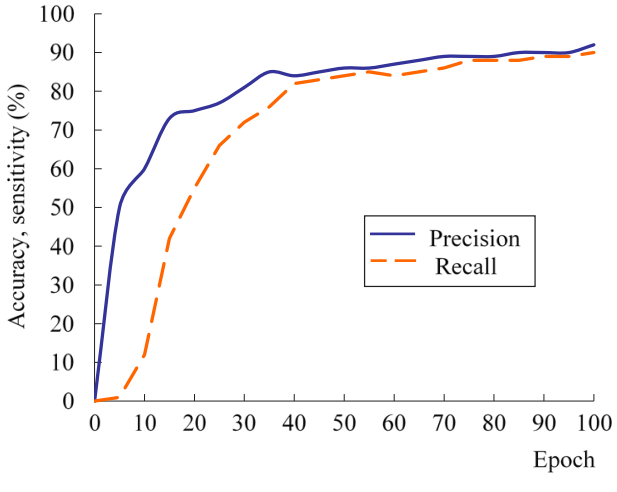


Fig. 12. Graphs of change in accuracy (precision) and sensitivity (recall) using the test sample depending on the epoch for the AlexNet 2012 model

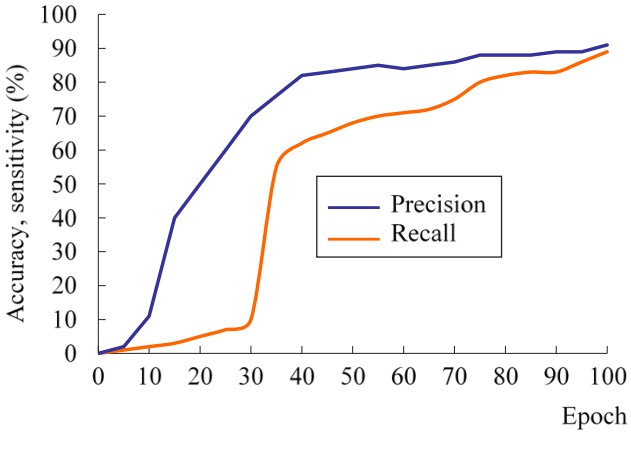


Fig. 13. Graphs of changes in the accuracy (precision) and sensitivity (recall) for the test sample depending on the epoch for the VGG-S 2014 model

It is seen in Fig. 14 that accuracy with the test sample stabilizes after the 30th epoch and varies in the range from 77 % to 89 %. Sensitivity ranges from 70 % to 87 % for the VGG-F 2014 model after the 60th epoch.

Summary results of estimating accuracy and sensitivity of the DCNN models under consideration are shown in Table 5.

Table 5

Estimation of accuracy and sensitivity of the models

Model	Accuracy (precision), %	Sensitivity (recall), %
AlexNet 2012	92	90
VGG-S 2014	90	89
VGG-F 2014	89	87

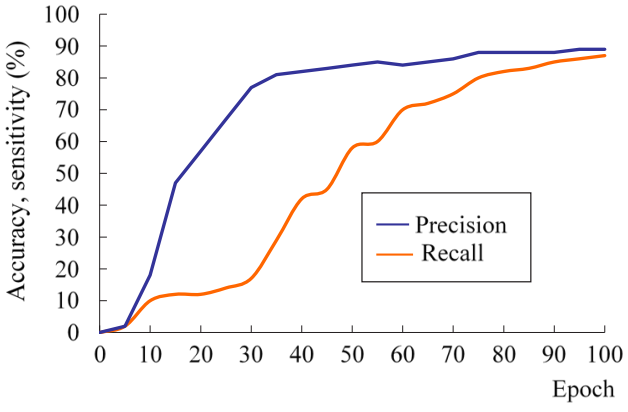


Fig. 14. Graphs of changes in the accuracy (precision) and sensitivity (recall) with the test sample depending on the epoch for the VGG-F 2014 model

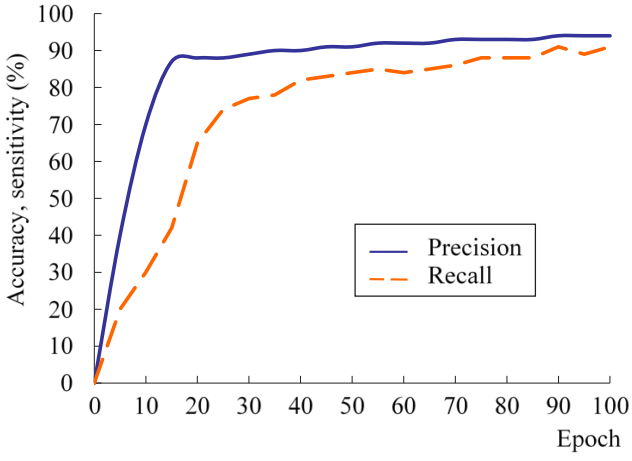


Fig. 15. Graphs of change in the accuracy (precision) and sensitivity (recall) with the test sample depending on the epoch for the proposed AlexVisDrone model

Analysis of Table 5 shows that the AlexNet 2012 model has demonstrated the best accuracy (92 %) and sensitivity (90 %). The object recognition error for this model with the ImageNet image set was 15% with an accuracy of 85 %. As a result, the recognition accuracy increased by 7 % confirming that the choice of the AlexNet 2012 DCNN architecture and training parameters was correct.

5. 2. Estimation of effectiveness of object image recognition by the proposed AlexVisDrone model

VisDrone 2021 image set was used in the evaluation of the effectiveness of image recognition by AlexNet 2012 model and its proposed AlexVisDrone modification. This set of images consists of 400 video files with 265,228 frames and 10,209 images taken with the camera installed on the UAV. The peculiarity of this set consisted in that it was made up under different lighting and weather conditions. 200 images from the VisDrone 2021 set were prepared for the training sample for each of 10 classes. 100 images were prepared for the test sample for each class. In total, 1000 object images were used for testing. The training and test sample type: a 227×227 JPEG image.

As noted earlier, to solve the problem of object recognition in aerial photographs in 10 classes and improve the effectiveness of object recognition, the final fully connected layer was modified from 1000 to 10 neurons and the modified AlexNet 2012 model was additionally trained in two stages. Additional training was performed with a set of images prepared from aerial photographs (par. 5. 1) at stage 1 and with a set of VisDrone 2021 images (China) at stage 2.

The study was conducted with the use of optimal values of the experimentally determined parameters (Table 4):

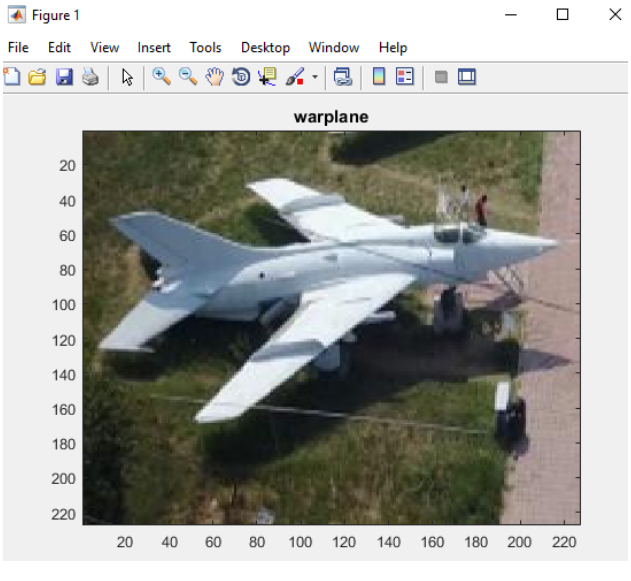
– training rate: 0.0001;
– training duration (number of epochs): 100;
– optimization algorithm: SGD;
– images from the set: VisDrone 2021;
– pretrained model: AlexNet 2012 DCNN.

As a result, a new model under the proposed name of AlexVisDrone was obtained. The results of checking its effectiveness are given in Fig. 15.

It can be seen from Fig. 15 that accuracy with the test sample for AlexVisDrone model has stabilized after the 15th epoch. Accuracy and sensitivity reached 94 % and 91 %, respectively, upon completion of the 100th epoch.

Fig. 16 shows a variant of the application of AlexVisDrone model.

a



b

Fig. 16. A variant of the model application: *a* – a snapshot from VisDrone2021 set; *b* – recognized image of the object (warplane) by the proposed AlexVisDrone model in MATLAB R2020b mathematical modeling environment

The new AlexVisDrone model was compared with the well-known models VGG-S 2014 and VGG-F 2014. To assess convergence, adequacy, and reliability of AlexVisDrone model, 140 images were used as a test sample for each of 10 classes from VisDrone2021 set.

Convergence. The convergence assessment shows that the choice of architecture and training parameters of the neural network was correct. The DCNN demonstrates convergence provided that the error decreases with each epoch. To provide convergence, training parameters and neural network architecture were selected. Fig. 17 shows the estimation of convergence for AlexVisDrone with the use of the test set.

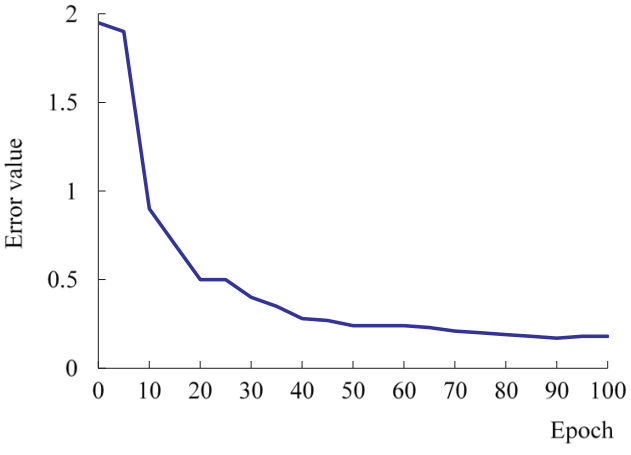


Fig. 17. Estimation of convergence for the proposed AlexVisDrone model

Analysis of Fig. 17 shows that the proposed DCNN AlexVisDrone has good convergence.

Adequacy. DCNN is adequate if the training results are very close to the nearest values (value) which is a necessary condition for the existence of some kind of regularity (a dependence between the output and input data) that can be implemented by the neural network.

The most preferred way of testing the DCNN model for adequacy involves the comparison of its results with those of known models.

The estimation results obtained with the test sample are shown in Table 6.

Table 6

The results of estimating accuracy and sensitivity of the models

Model	Accuracy (precision), %	Sensitivity (recall), %
VGG-S 2014	92	90
VGG-F 2014	90	89
Proposed AlexVisDrone model	94	91

It is seen from Table 6 that the developed AlexVisDrone model shows the highest accuracy (94 %) and sensitivity (91 %). Convergence of the results of estimating accuracy and sensitivity of the models (Tables 5, 6), as well as comparison of the results with known models (VGG-S, VGG-F), characterize adequacy of the developed model.

Reliability. Reliability of the proposed AlexVisDrone model is confirmed by the fact that the obtained results (recognition accuracy, sensitivity) are close (for the images prepared from aerial photographs (Table 5) and the VisDrone 2021 set (Table 6)) under chosen experimental conditions, limitations and assumptions.

The reliability of the results obtained is confirmed by the correctness of the use of the mathematical apparatus, as well as by the results of applying the proposed AlexVisDrone model to solving the practical problem of classifying the objects in aerial photographs.

Thus, the reliability of the proposed model is confirmed by the following:

– validity of the choice of initial data, basic constraints, and assumptions (Section 4);

– testing it with various training and test sets of images (ImageNet, VisDrone 2021 prepared from aerial photographs).

The object recognition error for AlexNet 2012 model with ImageNet image set was 15 % with an accuracy of 85 %. As a result, the developed AlexVisDrone model has shown a 9 % higher recognition accuracy.

6. Discussion of the results obtained in the study of recognizing the object images in aerial photographs using DCNN

It was proposed to use a trained DCNN to recognize images of military objects in aerial photographs. To improve effectiveness, the output fully connected layer was rejected from 1,000 to 10 neurons and additional two-stage training of the resulting model was performed: with a set of images prepared from aerial photographs (Fig. 9) at stage 1 and with a set of VisDrone 2021 images (Fig. 4) at stage 2. A new model under the proposed name of AlexVisDrone was obtained by selecting optimal parameters (speed (step) of training, number of epochs) (Fig. 2).

Application of the proposed model makes it possible to:

– automate the process of recognition of objects in aerial photographs;

– improve the accuracy of object recognition in images (in comparison with AlexNet 2012) from 7 % (for images from aerial photographs, Fig. 9) to 9 % (for the VisDrone 2021 set, Fig. 4) which confirms that the choice of the neural network architecture and training parameters was correct.

This AlexVisDrone model is a DCNN (Fig. 2).

Limitations of the proposed model:

– objects in aerial photographs were classified within 10 classes;

– the nabla function for the sigmoidal (logistic) activation function used in the work was reduced to the usual difference;

– orientation of objects in images was not taken into account;

– resolution of images for classification of objects was 227×227×3 (determined by AlexVisDrone DCNN feature, Fig. 5);

– translational invariance of the DCNN was not taken into account;

– aerial photographs were taken in the visible range during the daytime.

An important study issue consists in the control of "dead" zones of the activation functions. Two activation functions were used in the proposed model (as in the base one): ReLU (4) and Sofmax (7) [20, 21]. Mathematical

expressions, descriptions, and features of the application are given in Section 4. Estimation of the "dead" zone (percentage of zero values) was carried out after using ReLU for the first convolutional layer (*Conv* 1) of the proposed AlexVisDrone model. This model peculiarity consists in that the "dead" zone (percentage of zero values) will be different for each input image. The absence or presence of "dead" zones in the activation functions (after the activation functions) was estimated by counting the elements with a zero value for the case when the "war-plane" image was fed to the input of the neural network (Fig. 4). The "dead" zone in this image amounted to 9 %. After the maximum joining (*Max pool*) operation, there were 2 % of zero values. In fact, the application of *Max pool* operation after ReLU minimizes the percentage of zero values. To solve the "dead" zone problem, it is necessary to study the use of other "new" ReLUs: Leaky ReLU, Parametric ReLU, Randomized ReLU.

The computational complexity of the known neural network models [11–18] is one of their disadvantages. A solution to this problem can be found in the use of direct (Kronecker) penetrating product of matrices for the analytical description of operations performed in a concrete DCNN layer (expressions (10)–(24)). The proposed approach makes it possible to:

– formalize the model of the neural network layers;

– unify formalization of the description of neural networks of various structures and complexity;

– reduce computational costs;

– improve the speed of the DCNN (CNN).

One of the ways to avoid retraining of the proposed AlexVisDrone model (Fig. 2) and the AlexNet 2012 model (Fig. 1) consists in improving the training images. The first approach: rotation in a horizontal plane and cropping (Fig. 6). The second approach: varying brightness of the image RGB channels [21]. The study proposes a combined approach to improving the training images for neural networks by rotating in a horizontal plane and changing the brightness of the RGB image channels (Fig. 7) which makes it possible to:

– increase number of images in the training sample;

– adapt (train) the neural network to different orientations of the object image;

– avoid the DCNN retraining.

The proposed model limitations:

– it is adapted for object recognition in aerial photographs in ten classes;

– the DCNN is trained using digital images of objects of high contrast and definition obtained from aerial photographs taken from UAVs (Fig. 9) and a set of VisDrone 2021 images (Fig. 4);

– the photographs were taken in the summer daytime. Therefore, high accuracy and sensitivity of object recognition were obtained (Tables 5, 6). For other types of object images (different shooting conditions), accuracy and sensitivity of recognition in different classes may vary which requires additional study.

To further develop the proposed model, the following is planned:

– evaluate accuracy and sensitivity of the improved model for different conditions of object recognition;

– improve accuracy, sensitivity, and speed of the DCNN and reduce computation amount;

– study the proposed architecture with other activation functions (Leaky ReLU, Parametric ReLU, Randomized ReLU, etc.);

– develop a method of detecting objects in aerial photographs with subsequent cropping the object images of required resolution;

– develop a method of detecting objects in video streams that were shot by the UAV optical system.

7. Conclusions

1. Indicators of effectiveness of AlexNet 2012, VGG-S, VGG-F models have been studied. The object recognition error for the AlexNet 2012 model with the ImageNet set of images amounted to 15 % with an accuracy of 85 %. The effectiveness of the AlexNet 2012, VGG-S, VGG-F models was tested on basis of the images prepared from aerial photographs (1400 images for a training sample, 700 images for a test sample). It was found that the best accuracy (92 %) and sensitivity (90%) were shown by AlexNet 2012 model at a training rate of 0.0001 when the CGD optimization algorithm was applied. The VGG-F model showed the smallest values (for images prepared from aerial photographs) in accuracy (89 %) and sensitivity (87 %).

2. Effectiveness of the proposed AlexVisDrone model (based on the images prepared from the VisDrone 2021 set) was assessed. Accuracy and sensitivity (94 % and 91 %, respectively) were obtained with the proposed AlexVisDrone model. The accuracy of this model exceeded 90% on completion of the 90th epoch. The model had the best performance indicators in comparison with the VGG-S, VGG-F models. The VGG-F had the lowest accuracy (90 %) and sensitivity (89 %). The developed AlexVisDrone model has shown a 9 % increase in recognition accuracy in comparison with the AlexNet 2012 model (trained with the ImageNet set). The obtained performance indicators of the AlexVisDrone model make it possible to assert the correctness of the choice of the DCNN architecture and its training parameters. It is advisable to use this model at the ground UAVC control stations when processing aerial photographs taken from the UAVs, in robotic systems, in video surveillance complexes and when designing the systems for unmanned vehicles.

References

1. Yang, X., Lin, D., Zhang, F., Song, T., Jiang, T. (2019). High Accuracy Active Stand-off Target Geolocation Using UAV Platform. 2019 IEEE International Conference on Signal, Information and Data Processing (ICSIDP). doi: http://doi.org/10.1109/icsidp47821.2019.9172919
2. Pospelov, B., Andronov, V., Rybka, E., Krainiukov, O., Maksymenko, N., Meleshchenko, R. et. al. (2020). Mathematical model of determining a risk to the human health along with the detection of hazardous states of urban atmosphere pollution based on measuring the current concentrations of pollutants. Eastern-European Journal of Enterprise Technologies, 4 (10 (106)), 37–44. doi: http://doi.org/10.15587/1729-4061.2020.210059
3. Chernukha, A., Teslenko, A., Kovalov, P., Bezuglov, O. (2020). Mathematical Modeling of Fire-Proof Efficiency of Coatings Based on Silicate Composition. Materials Science Forum, 1006, 70–75. doi: http://doi.org/10.4028/www.scientific.net/msf.1006.70

4. Vambol, S., Vambol, V., Kondratenko, O., Suchikova, Y., Hurenko, O. (2017). Assessment of improvement of ecological safety of power plants by arranging the system of pollutant neutralization. Eastern-European Journal of Enterprise Technologies, 3 (10 (87)), 63–73. doi: http://doi.org/10.15587/1729-4061.2017.102314
5. Vambol, S., Vambol, V., Kondratenko, O., Koloskov, V., Suchikova, Y. (2018). Substantiation of expedience of application of high-temperature utilization of used tires for liquefied methane production. Journal of Achievements in Materials and Manufacturing Engineering, 2 (87), 77–84. doi: http://doi.org/10.5604/01.3001.0012.2830
6. Teslenko, A., Chernukha, A., Bezuglov, O., Bogatov, O., Kunitsa, E., Kalyna, V. et. al. (2019). Construction of an algorithm for building regions of questionable decisions for devices containing gases in a linear multidimensional space of hazardous factors. Eastern-European Journal of Enterprise Technologies, 5 (10 (101)), 42–49. doi: http://doi.org/10.15587/1729-4061.2019.181668
7. Semko, A., Rusanova, O., Kazak, O., Beskrovnaya, M., Vinogradov, S., Gricina, I. (2015). The use of pulsed high-speed liquid jet for putting out gas blow-out. The International Journal of Multiphysics, 9 (1), 9–20. doi: http://doi.org/10.1260/1750-9548.9.1.9
8. Pospelov, B., Andronov, V., Rybka, E., Popov, V., Semkiv, O. (2018). Development of the method of frequencytemporal representation of fluctuations of gaseous medium parameters at fire. Eastern-European Journal of Enterprise Technologies, 2 (10 (92)), 44–49. doi: http://doi.org/10.15587/1729-4061.2018.125926
9. Pospelov, B., Rybka, E., Meleshchenko, R., Borodych, P., Gornostal, S. (2019). Development of the method for rapid detection of hazardous atmospheric pollution of cities with the help of recurrence measures. Eastern-European Journal of Enterprise Technologies, 1 (10 (97)), 29–35. doi: http://doi.org/10.15587/1729-4061.2019.155027
10. Slyusar, V., Protsenko, M., Chernukha, A., Gornostal, S., Rudakov, S., Shevchenko, S. et. al. (2021). Construction of an advanced method for recognizing monitored objects by a convolutional neural network using a discrete wavelet transform. Eastern-European Journal of Enterprise Technologies, 4 (9 (112)), 65–77. doi: http://doi.org/10.15587/1729-4061.2021.238601
11. Sermanet, P., Kavukcuoglu, K., Chintala, S., Lecun, Y. (2013). Pedestrian Detection with Unsupervised Multi-stage Feature Learning. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 3626–3633. doi: http://doi.org/10.1109/cvpr.2013.465
12. Lu, X., Ma, C., Ni, B., Yang, X. (2021). Adaptive Region Proposal With Channel Regularization for Robust Object Tracking. IEEE Transactions on Circuits and Systems for Video Technology, 31 (4), 1268–1282. doi: http://doi.org/10.1109/tcsvt.2019.2944654
13. Li, J., Weinmann, M., Sun, X., Diao, W., Feng, Y., Fu, K. (2021). Random Topology and Random Multiscale Mapping: An Automated Design of Multiscale and Lightweight Neural Network for Remote-Sensing Image Recognition. IEEE Transactions on Geoscience and Remote Sensing, 1–17. doi: http://doi.org/10.1109/tgrs.2021.3102988
14. Zhang, Z., Zhang, L., Wang, Y., Feng, P., He, R. (2021). ShipRSImageNet: A Large-Scale Fine-Grained Dataset for Ship Detection in High-Resolution Optical Remote Sensing Images. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 14, 8458–8472. doi: http://doi.org/10.1109/jstars.2021.3104230
15. Fan, J., Lee, J., Lee, Y. (2021). Image Classification Using Fusion of Multiple Neural Networks. 2021 36th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC). doi: http://doi.org/10.1109/itc-cscc52171.2021.9501468
16. Wu, C., Shao, S., Tunc, C., Hariri, S. (2020). Video Anomaly Detection using Pre-Trained Deep Convolutional Neural Nets and Context Mining. 2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA). doi: http://doi.org/10.1109/aiccsa50499.2020.9316538
17. Lian, D., Hu, L., Luo, W., Xu, Y., Duan, L., Yu, J., Gao, S. (2019). Multiview Multitask Gaze Estimation With Deep Convolutional Neural Networks. IEEE Transactions on Neural Networks and Learning Systems, 30 (10), 3010–3023. doi: http://doi.org/10.1109/tnnls.2018.2865525
18. Scott, G. J., Marcum, R. A., Davis, C. H., Nivin, T. W. (2017). Fusion of Deep Convolutional Neural Networks for Land Cover Classification of High-Resolution Imagery. IEEE Geoscience and Remote Sensing Letters, 14 (9), 1638–1642. doi: http://doi.org/10.1109/lgrs.2017.2722988
19. Li, H., Li, J., Han, X. (2019). Robot Vision Model Based on Multi-Neural Network Fusion. 2019 IEEE 3rd Information Technology Networking Electronic and Automation Control Conference (ITNEC), 2571–2577. doi: http://doi.org/10.1109/itnec.2019.8729210
20. Knysh, B., Kulyk, Y. (2021). Improving a model of object recognition in images based on a convolutional neural network. Eastern-European Journal of Enterprise Technologies, 3 (9 (111)), 40–50. doi: http://doi.org/10.15587/1729-4061.2021.233786
21. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60 (6), 84–90. doi: http://doi.org/10.1145/3065386
22. Sugoniaev, A. (2018). Iskusstvennye neironnye seti. Funktsiia poter. Saint-Petersburg. Available at: https://ppt-online.org/338631
23. Slyusar, V. (2021). Neural Networks Models based on the tensor-matrix theory. Problems of the development of promising micro- and nanoelectronic systems (MNS-2021), 23–28. doi: http://doi.org/10.31114/2078-7707-2021-2-23-28
24. Slyusar, V. I. (1999). A family of face products of matrices and its properties. Cybernetics and Systems Analysis, 35 (3), 379–384. doi: http://doi.org/10.1007/bf02733426
25. Shchegolev, A. (2020). Development of an element base for superconducting artificial neural networks based on macroscopic quantum effects. Moscow. Available at: https://istina.msu.ru/download/321964642/1kGL1g:Naktuv1mMyFA6kEvKTBPEfux51U/
26. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15 (56), 1929–1958. Available at: http://jmlr.org/papers/v15/srivastava14a.html