Розроблено модифікації простого генетичного алгоритму для розпізнавання образів. У запропонованій модифікації Альфа-Бета на етапі відбору особин до нової популяції особини ранжуються за показником пристосованості, далі випадковим чином визначається кількість пар – певна кількість найпристосованіших особин, та стільки ж найменш пристосованих. Найпристосованіші особини формують підмножину B, найменш пристосовані – підмножину W. Обидві підмножини входять в множину пар V. Число особин, що можуть бути обрані в пари, знаходиться в діапазоні 20–60 % від загальної кількості особин. У модифікації Альфа-Бета фіксована в порівнянні з оригінальною версією простого генетичного алгоритму було додано можливість виникнення двох мутацій, додано фіксовану точку схрещення, а також змінено відбір особин для схрещення. Це дозволяє підвищити показник точності у порівнянні з базовою версією простого генетичного алгоритму. У модифікації Фіксована встановлено фіксовану точку схрещення. В схрещенні приймає участь половина генів – гени що відповідають за кількість нейронів на шарах, значення інших генів завжди передаються нащадкам від однієї з особин. Також, на етапі мутації випадковим чином відбуваються мутації з використанням методу Монте-Карло.

Розроблені методи програмно реалізовано для вирішення задачі розпізнавання учасників дорожнього руху (автомобілів, велосипедів, пішоходів, мотоциклів, вантажівок). Також було проведено порівняння показників використання модифікацій простого генетичного алгоритму та визначено кращий підхід вирішення задачі розпізнавання учасників дорожнього руху. Було встановлено, що розроблена модифікація Альфа-Бета показала кращі результати у порівнянні з іншими модифікаціями при вирішенні задачі розпізнавання учасників дорожнього руху. При застосуванні розроблених модифікацій отримано наступні показники точності: Альфа-Бета – 96,90 %, Альфа-Бета фіксована – 95,89 %, фіксована – 85,48 %. Крім того, при застосуванні розроблених модифікацій скорочується час підбору параметрів нейромоделі, зокрема при використанні модифікації Альфа-Бета витрачається лише 73,9 % часу базового методу, при використанні модифікації Фіксована – 91,1 % часу базового генетичного методу

Ключові слова: розпізнавання образів, генетичний алгоритм, еволюційний алгоритм, нейронні мережі, Python, OpenCV, Keras

UDC 004.93
DOI: 10.15587/1729-4061.2019.164789

DEVELOPMENT OF THE MODIFIED METHODS TO TRAIN A NEURAL NETWORK TO SOLVE THE TASK ON RECOGNITION OF ROAD USERS

I. Fedorchenko
Senior Lecturer*
E-mail: evg.fedorchenko@gmail.com
A. Oliinyk
PhD, Associate Professor*
E-mail: olejnikaa@gmail.com
A. Stepanenko
PhD, Associate Professor*
E-mail: alex@zntu.edu.ua
T. Zaiko
PhD, Associate Professor*
E-mail: nika270202@gmail.com
S. Shylo
Senior Lecturer
Department of Electrical and Electronic Apparatus**
E-mail: sergey.shilo@gmail.com
A. Svyrydenko
Software Developer
E-mail: antonsvyrydenko@gmail.com
*Department of Software Tools**
**Zaporizhzhya National Technical University
Zhukovskoho str., 64,
Zaporizhzhya, Ukraine, 69063

1. Introduction

The number of fields involving pattern recognition is constantly growing, it is associated with the development of technologies, methods, software solutions, libraries, computational equipment, as well as with the need to automate or control processes without a human.

One of such fields is road safety, which includes many narrow areas: recognition of license plates, determining busiest streets, recognition of vehicles in unmanned transport, etc.

However, many problems related to recognition quality have remained unresolved up not now. For example, the system Autopilot Tesla has been found to demonstrate a vulnerability – in most cases the system does not recognize a bicycle, it defines a bike as a person or a small car; the system, accordingly, could make a decision the result of which might pose a threat to the life of a cyclist [1].

There are also known vulnerabilities in recognition of motorists and vehicles in such systems as Mazda Smart City Brake Support [2], the autopilot Waymo [3].

One can say that it is a rather relevant task to recognize bicycles, as well as other motorists. Recognition of road users can be used in the system of control and/or road safety, or to study the relevance of constructing specialized bicycle lanes, etc.

2. Literature review and problem statement

Paper [4] reports a research into solving the task on recognition of bicycles in a video stream by combining the following approaches: the method of Histogram of Oriented Gradients (HOG) [5, 6], the method of Support Vector Machine (SVM) [7], a cascade classifier (Viola-Jones object detection) [8, 9]. The paper reports results from the process of bicycle recognition under various weather conditions and argues that the approach, developed in study [4], can be used in real-time recognition systems.

However, using a combination of methods of HOG and SVM is not very effective, as it makes it possible to identify sufficiently large objects that are not always present on video in real time, and because one needs quite a lot of time to compute complex parameters. Th use of cascade classifier is more efficient in terms of recognition time. However, a given method has errors in recognition – due to the application of different scale and size of the scanning window, a single object can be recognized as two objects.

In general, the low image quality, weather conditions, as well as different lighting, significantly complicate the use of methods [4–9]. The approach, developed in [4], is quite complex and challenging while every method [5–9] requires substantial temporal and computing resources that affects the results obtained and the effectiveness of using such methods in practice. The use of neural networks might prove to be a more reasonable approach to implementing bicycle recognition.

Thus, paper [10] reports a study into analysis of flow of vehicles acquired from CCTV images (video surveillance systems), based on solving the task of vehicle recognition using the neural networks Faster R–CNN (faster region based convolutional neural network) [11] and SSD (single shot multibox detector) [12].

In general, the SSD networks can quickly execute recognition, but they are often mistaken while recognizing small objects, while Faster R–CNN work slower, but capable to provide greater accuracy.

Paper [12] tested the above neural networks trained on images of low quality and under different weather conditions. The result reported in the study point to the fact that the examined neural networks were not able to recognize all vehicles; in this case, bad weather conditions (rain, snow, fog) decrease the indicator of accuracy dramatically. Thus, when training neural networks for vehicle recognition one should have used a more diverse learning sample. In addition, the study describes the vehicle recognition only, although the concept of «traffic» covers several modes of transportation (motorcycles, trucks, etc.).

Work [13] considered a solution to the task on vehicle recognition by training a convolutional neural network [14], based on the related error of learning and the samples prone to errors. That is, when training a convolutional neural network, it is proposed to use learning mistakes within current stage as a training dataset at the next stage. The authors compared recognition by the method of histogram of oriented gradients and by convolutional neural networks, one of which was trained in a standard way, and the second was proposed by the authors, the result being that the approach, suggested in [13], demonstrated better performance (specifically, the recognition accuracy was 83.91 %).

However, there are no examples regarding the recognition of vehicles and, given a small size of the training sample that was used in the study, one cannot say with full confidence that there has been a significant improvement in the learning by a convolutional neural network.

Paper [15] considered a solution to the task on recognition every pedestrian among a crowd. The main purpose of the study was to recognize a pedestrian at black and white images in a variety of situations, pedestrian detection in an image, even at partial overlay, and to localize him/her at high accuracy. In addition, the goal was to determine the number of pedestrians shown in the image.

The approach developed in a given article is based on the scale-independent extension of the Implicit Shape Model (ISM). The authors were able to implement all the set goals, although a pedestrian recognition accuracy is equal to 71.3 %: it is not satisfactory. In addition, it is worth noting that the developed method operates only on images. However, the method has a great potential and could, as a result of further advancement, evolve towards pattern recognition and tracking.

Paper [16] considered a solution to the task on pedestrian recognition by using a combination of multiple neural networks. The proposed system consists of two subsystems: the main pedestrian detection system for generating all detections and the system for semantic segmentation to improve the results. The pedestrian detection system in turn consists of a generator of «candidates» to pedestrians and a classification system. In order to provide more information about coordinates of the object for network classification, it was proposed to use a new method of «soft attributes», which relates an object to all classes. To implement the system of classification, the idea was introduced of «mixed learning». In addition, there is the procedure of «mild rejection» to combine the findings of all classification networks and a semantic segmentation network with the conclusion by the network that generates «candidates».

The recognition of pedestrians was tested on four popular image sets: the sets Caltech Pedestrian, INRIA, ETH, KITTI. The first three sets yielded the highest indicators of recognition accuracy. A considerable performance speed of the developed approach was also noted.

Paper [17] considered pedestrians recognition using a combination between the method of histogram of oriented gradients and the method of support vectors and a convolutional neural network. The aim of the work was to construct semantic regions of interest, in order to acquire a foreground object, to reduce errors associated with the background recognition errors. First, by using a convolutional neural network, trained on the set Caltech Pedestrian, they generated a heat map based on the input image. Next, semantic regions of interest are extracted from the heat map using a morphological image processing. Finally, semantic regions of interest parse the entire image into a background and a foreground, to facilitate the work of detectors that would make a decision.

It is noted that using the semantic regions of interest improves detectors' performance to a certain degree. However, the application of the combination of HOG and SVM methods is not optimal, because it does not make it possible to identify sufficiently large objects that are not always present on video in real time, and because one needs quite a lot of time calculating complex settings.

Paper [18] addressed the implementation of a recognition system for vehicles belonging to 7 classes (a motorcycle, a car, a pickup, a bus, a truck, a truck with trailer, a truck with multiple trailers) on CCTV video. The implementation employed a trained Deep Convolutional Neural Network (DCNN).

The proposed system algorithmically consists of two tasks: localization and classification. Localization is executed first, using the generation of regions independent from the class, for each frame, then a deep convolutional neural network is applied to acquire the descriptions of attributes for each region. Finally, by using the method of Support Vector Machines (SVM) for each region, the acquired descriptions of attributes are compared with templates, followed by the conformity assessment and classification. A recognition accuracy of vehicles is different for each class, but the overall accuracy indicator is within 92–95 %. An accuracy recognition indicator decreases substantially at heavy traffic, rain and fog, at low video quality.

The disadvantage of this development is the use of a deep convolutional neural network since its computation requires significant computing resources, which is not acceptable. The use of a given neural network is possible in the presence of a powerful processor (CPU), or the graphics processing unit (GPU) NVIDIA, in a tandem with such programming environments of artificial intelligence as CUDA, Caffe, Torch. In addition, the system revealed errors in recognition, at a small scale, when an object from a particular class is recognized as an object from another class, for example, a «truck» is recognized as a «truck with a trailer», etc.

Article [19] describes a method of «an end-to-end» recognition of moving objects in a video stream in real time, their classification according to the specified categories in line with the properties based on images and their further monitoring. Recognition of moving objects is carried out by using a pixel difference between consecutive frames of an image. The classification metric employs these objects with a temporal constraint for coherence, in order to classify them for three categories: a person, a vehicle, or a background. Following the classification, objects are tracked by merging temporal differences and pattern-matching.

The two key elements that make the system robust are the classification system based on temporal coherence, and the tracking system based on a combination between the temporal difference and correlation matching. The system effectively integrates basic knowledge of the subject area about the classes of objects with statistical indicators in a temporal domain for the classification of target objects.

The advantage of a given method is that it allows continuous tracking, despite the occlusion and termination of an object's movement, prevents the «drift» of templates onto a background texture, and provides for reliable tracking, without the need for a special filter, such as a Kalman filter .

The main disadvantage of this method was an incorrect classification, which occurred for the case when several objects were close to each other. In addition, small objects are often not recognized as being stable over time.

Work [20] addresses the task on detection and tracking of moving objects in a video stream acquired from a mobile air platform. The examined method is based on the graphical representation of moving objects, which makes it possible to display and maintain a dynamic template of each moving object by ensuring their temporal coherence. A dynamic template, along with a graphical representation that is used in a given approach, makes it possible to characterize the trajectories of objects as an optimal path in a graph. The proposed tracker makes it possible to solve partial occlusions, to stop and move, even in very difficult situations. The paper reports results based on a series of different real sequences. The purpose of tracking and detection of objects is to establish a match between the objects or parts of objects in consecutive frames and to derive temporal information about such objects as a trajectory, position, speed, and direction. Tracking the object, detected in a frame, by a video frame is an important and difficult task.

The disadvantage of that work is that one needs to consider if the total video signal contains a very large amount of information, it is important to acquire and process only a small share of relevant data, especially in cases where it is necessary to obtain performance at a frame rate in real time.

Paper [21] considered the use of Switchable Deep Network (SDN) for recognition of pedestrians. A given work resolves actual tasks on the recognition of pedestrians, specifically errors in the recognition of a background, a large number of variations in the physical appearance of pedestrians as they change positions and other factors. One of the common problems for the systems of pedestrian recognition relates to that the system can recognize a certain object to be a pedestrian. For example, the shape and appearance of a «tree trunk» or an «iron pillar» at a certain point of view are similar to pedestrians. Thus, to solve this task, a given work proposed using the Restricted Boltzmann Machine (RBM) for recognition of pedestrians. The application of SDN improves a standard convolutional neural network by adding several restrictive layers that are created with the new restricted Boltzmann machine, which makes it possible, as a result, to automatically learn both the low-level features and the high-level parts (for example, «head», «feet», etc.) in a pedestrian.

However, the gain in productivity owing to the Switchable Deep Network was insignificant. The developed method needed quite a lot of time for recognition of complex objects, thus the issue on how to recognize a pedestrian to achieve optimal performance has remained unresolved.

A pedestrian recognition task has some specific features. One of these features is the need to correctly identify pedestrians of different size. Different size of pedestrians may be predetermined both by a distance from the detector and by different growth and physique of most people (for example, a child and a basketball player). A possibility to recognize multidimensional objects could be significantly improved by a detector from [22]. For example, when using a detector in a car, the correct detection of pedestrians not only close to it, but at a greater distance as well, will provide for a better control over a road situation in order to make timely decisions.

Most often, the detector [22] calculates the scalar product between the trained pattern (sensitive field) and the recognized region of the image. For correct operation of the detector, dimensionality of the trained template should match the dimensions of the object to be recognized. There are two basic methods for solving the task on recognition of multi-scale objects.

A first one, described in work [23], implies training a single classifier and resizing the input images, so that the classifier is applicable for all possible sizes. This method is the most accurate, but it requires a large amount of computations, because it requires calculation of attributes at each change in the size of an input image.

The disadvantage of the first method is that the size of a candidate's region will change to the size that the network works with and, accordingly, the attributes will be calculated not for the entire image, but only for a certain region.

A second method [24] implies training several classifiers that will be suitable to recognize all sizes of objects at a con-

stant size of an input image. This method makes it possible to avoid repeated computation of attributes, but the quality of recognition is quite low, because each of the possible size of the object would require a separate classifier.

The disadvantage of the second method is that there is a strong divergence in the size of the recognized object and a sample.

An analysis of the scientific literature [4, 10, 13, 15–18] allows us to assert that it is a rather relevant task to undertake a study aimed at improving recognition accuracy; it has also revealed the following unresolved problems:

– human occlusion. As people appear against various and unpredictable backgrounds, occlusion can occur at any time. Thus, to achieve high performance when detecting a person in the process of recognition, the problem of occlusion should be effectively treated;

– human articulation. The appearance of a person can be extremely varied: due to a change in position, distance, or the point of camera's view. Thus, the recognition algorithms in the process of identifying a person must consider this aspect to make the system more reliable and accurate;

– background noise. Changes in context due to weather conditions, changes in lighting and complicated backgrounds are the most important reasons for the detection of a fault or a gap, which in some cases significantly reduces the accuracy of recognition as well;

– processing time. Processing time is a strict requirement for many real-world applications that work in real time. Modern approaches to human or vehicle recognition still need a clear improvement to satisfy this requirement and reduce the time of recognition.

Such problems in some cases significantly reduce the recognition accuracy, making it difficult to apply known methods in practice.

Existence of these deficiencies necessitates the development of evolutionary methods for constructing recognition models that would make it possible to recognize vehicles at higher accuracy over shorter time.

3. The aim and objectives of the study

The aim of this study is to develop mathematical support for solving the task on the recognition of road users (cars, bicycles, pedestrians, motorcycles, trucks) based on the modifications of a simple genetic neural network learning method. This will make it possible to recognize images of low quality and under different weather conditions.

To accomplish the aim, the following tasks have been set:

– to develop modifications of a simple genetic neural network learning method;

– to implement in software the developed modifications of a simple genetic method;

– to examine effectiveness of the developed modifications of a simple genetic method.

4. Development of modifications of a simple genetic neural network learning method. Determining the impact of parameters for training neural networks

We have proposed in the developed modifications of a simple genetic method, in order to improve the efficiency of evolutionary search, new heuristic procedures, specifically a possibility of the occurrence of two mutations has been added, particularly using a Monte Carlo method, and the operators of selection and crossbreeding have been modified. This makes it possible to increase the accuracy indicator compared to the basic version of a simple genetic algorithm. The proposed modifications of a simple genetic algorithm are as follows.

The proposed modification Alpha–Beta implies the selection for crossbreeding in each generation a different quantity of pairs for crossbreeding; in this case, one individual from the pair belongs to those most fit while another one to the least adapted individuals. In addition, there may randomly occur two mutations (basic and doubling) or a single mutation (basic): the Monte Carlo method produces a random number, 0 or 1. If it is 0, then one mutation appears, if it is 1 – there are two mutations.

The sequence of steps in a given modification of a simple genetic algorithm is similar to its basic version, but has some differences. At the stage of selection of individuals to the new population P_n the individuals are ranked in terms of fitness, then one randomly determines the number of pairs – a certain number of the most fit individuals, and the same number of those least adapted. The most fit individuals form the subset B, those least adapted – the subset W. Both subsets are included in the set of pairs V. The number of individuals that can be selected to pair is in the range of 20–60 % of the total number of individuals. Such characteristics for a range of individuals were selected as a result of numerical experiments and research into the developed modification of the genetic method. The rest of the new population P_n is obtained by crossbreeding the selected individuals (K).

$$P_n=(V, K_j), \quad (1)$$

where $j=1,..., (N-V)$.

In addition, in the proposed modification there may randomly occur two (μ_1, μ_2) or a single mutation (μ), whereas the basic version of the algorithm randomly produces one mutation. Moreover, in a situation where there are two mutations, one gene can mutate twice.

After applying the operator of mutation, descendants are included to the new generation P_n. Crossbreeding and mutation are repeated until a new generation P_n forms of size N (1).

In the second proposed modification Alpha-Beta fixed for crossbreeding in each generation one selected a different number of pairs for crossbreeding; in this case, within a pair, one individual belongs to the most fit individuals, and a second – to those least adapted. In addition, there may randomly occur two mutations (basic and doubling) or a single mutation (basic): the Monte Carlo method generates a random number, 0 or 1. If it is 0, then there is a single mutation, if it is 1 – there are two mutations. And we set a fixed point of crossbreeding– the crossbreeding involves the first half of genes – the genes the are responsible for the number of neurons in layers, values for other genes are always transmitted to the descendants by one of the individuals.

The sequence of steps of a given modification is similar to the steps of the modification Alpha-Beta, however, the operation of crossbreeding is different. We select randomly from the subsets B and W individuals M and F that have a specific set of genes:

$$M=\left(\left(n_1^1,\ldots,n_m^1\right),m^1,f_{act}^1,f_{opt}^1\right),$$

$$F=\left(\left(n_1^2,\ldots,n_m^2\right),m^2,f_{act}^2,f_{opt}^2\right).$$

Thus, the crossbreeding operator can be represented as follows:

$$C^* = \left(\left(\frac{1}{2}M \times \frac{1}{2}F\right), \frac{1}{2}F\right). \quad (2)$$

The result of individuals crossbreeding is two descendants K_1, K_2, which can be recorded as:

$$K_1 = \left(R\left[\left(n_1^1,\ldots,n_m^1\right),\left(n_1^2,\ldots,n_m^2\right)\right], m^2, f_{act}^2, f_{opt}^2\right),$$

$$K_2 = \left(R\left[\left(n_1^1,\ldots,n_m^1\right),\left(n_1^2,\ldots,n_m^2\right)\right], m^2, f_{act}^2, f_{opt}^2\right), \quad (3)$$

where R is the function of a random magnitude selection; m^2, f_{act}^2, f_{opt}^2 are the values for genes transmitted from individual F.

The developed modification Fixed implies that we established a fixed point of crossbreeding: the crossbreeding involves half the genes – the genes that are responsible for the number of neurons in layers, values for other genes are always transmitted to descendants by one of the individuals. In addition, at the stage of mutation there may randomly occur two mutations (basic and doubling) or a single mutation (basic): the Monte Carlo method generates a random number, 0 or 1. If it is 0, then there is a single mutation, if it is 1 – there are two mutations.

The sequence of steps in a given modification is similar to the sequence of steps in the modification Alpha-Beta, but has a certain difference – the selection of individuals to the new population occurs similar to the basic version of the genetic algorithm $P_n = T + L + K$, and crossbreeding occurs similar to the second modification (2), (3).

Thus, we have constructed modified genetic methods for the selection of parameters to train the neural networks – Alpha-Beta, Alpha-Beta fixed, Fixed.

5. Software implementation of the developed modifications of a simple genetic method

Software implementation of the developed modifications of a simple genetic method includes various applications: pattern recognition using genetic methods, a cascade classifier, training a neural network, testing the recognition by a neural network, selection of settings when training neural networks using modifications of a simple genetic algorithm (Fig. 1).

Software implementation of pattern recognition consists of several modules, interrelated by methods, so the principal diagram of the software can be represented in the following way (Fig. 2).

Software implementation of the selection of parameters for training learning neural networks using the modified genetic methods can be represented as follows (Fig. 3).

Software implementation of training a convolutional neural network can be represented in the following way (Fig. 4).

The developed software implementation is a set of programs for pattern recognition, which contains files (scripts, modules) for pattern recognition.

The main module of software is main.py, it calls the programs listed above. This module contains the following methods:

– the initUI – method for creating a software window;

– disable_buttons – method that limits the possibility to call programs;

– enable_buttons – method that enables the possibility to call programs;

– the error_message – method for displaying an error window;

– info_message – method for displaying a message window;

– cascade_button – method for calling a program that tests the detection of a cascade classifier;

– ga_button – method for calling a program to select parameters when training neural networks;

– test_button – method for calling a program that tests the recognition by a neural network;

– train_button – method for calling a program to train a neural network.

The pattern recognition software that employs a cascade classifier consists of two files: cascade_test.py and cascade.xml.

File cascade_test. py is the main file of the program that uses the cascade classifier for recognition, which reads the image and file of the cascade classifier, processes and recognizes an image, and displays the resulting image. The script employs the following methods:

– CascadeClassifier – method for reading a cascade classifier from the library OpenCV;

– imread – method for reading an image from the library OpenCV;

– datetime. now – method for determining current time. It is used to compute the time required for recognition;

– cvtColor – method for converting an image to another color space from the library OpenCV;

– detectMultiScale – method of object recognition from the library OpenCV;

– non_max_suppression – method that can reduce an error in recognition from the library imutils.

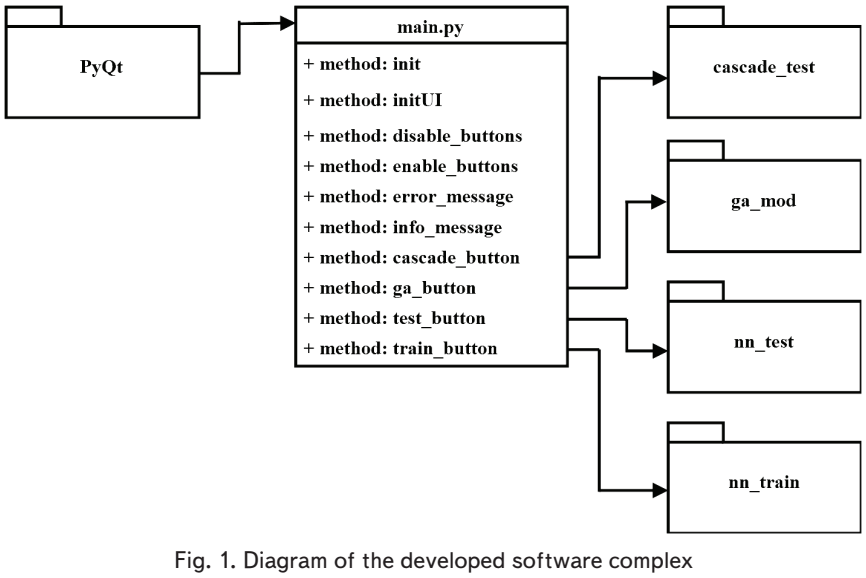

Fig. 1. Diagram of the developed software complex

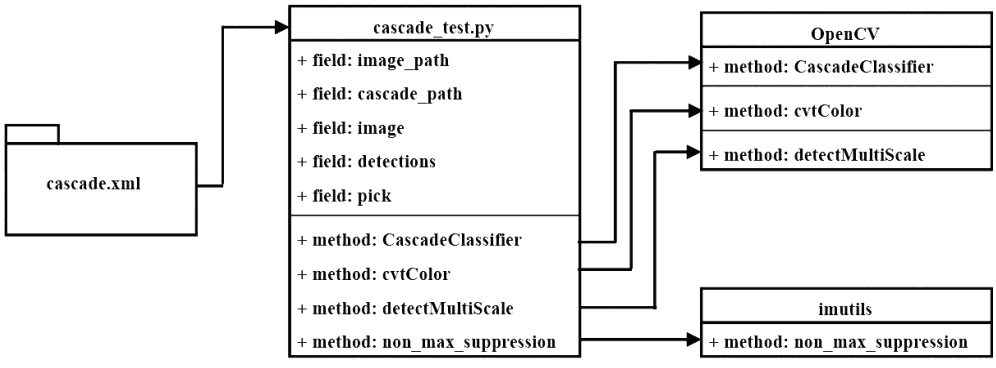

Fig. 2. Diagram of software for testing a cascade classifier

File cascade. xml – the trained cascade classifier for an object recognition.

The software to select settings when training a neural network consists of the following files: main.py, network.py, optimizer.py, train.py, in the container ga_mod.

File main.py is the main program file, initializing the values for parameters that will be used when training neural networks, calling the training of neural networks. The script has the following methods:

– train_networks – calls the method train in the script network.py and displays the progress of neural networks learning;

– avg_accuracy calculates the average accuracy indicator for the trained networks;

– generate_network calls methods to train neural networks and methods for selecting the parameters in the script optimizer.py;

– print_networks displays information about neural networks with the best indicators;

– main – initiates the values for parameters that will be used when training neural networks, and calls the method generate.

File network.py is intended to store information about a neural network. It included the following methods:

– init – initiates a network;

– random_network – fills the genes in a network with random values;

– create_set – assigns values of the generated genes to an appropriate network;

– train_network – calls the method for training neural networks train_and_score in the script train.py;

– print_network displays information about indicators of the trained neural network.

File optimizer.py contains the implementation of the modified simple genetic algorithm for the selection of optimal parameters for training a neural network. It contains the following methods:

– init – initiates parameters for selection of parameters;

– create_population – creates a population of neural networks;

– breed – runs the crossbreeding of two neural networks (modification Alpha-Beta);

– breed_mod2_3 – runs the crossbreeding of two neural networks (modifications Alpha-Beta fixed and Fixed);

– mutate – mutation operator is executed to the genes of a neural network (two mutations – basic and doubling);

– mutate_basic – mutation operator is executed to the genes of a neural network (one mutation – basic);

– evolve – selects neural networks for the next generation (modifications Alpha-Beta and Alpha-Beta fixed);

– evolve_mod3 – selects neural networks for the next generation (modification Fixed).

File train.py is intended to download a training sample and to train a neural network. It contains the following methods:

– custom – downloads a training sample;

– compile_model – creates the structure of a neural network;

– compile_evaluate – trains a neural network.

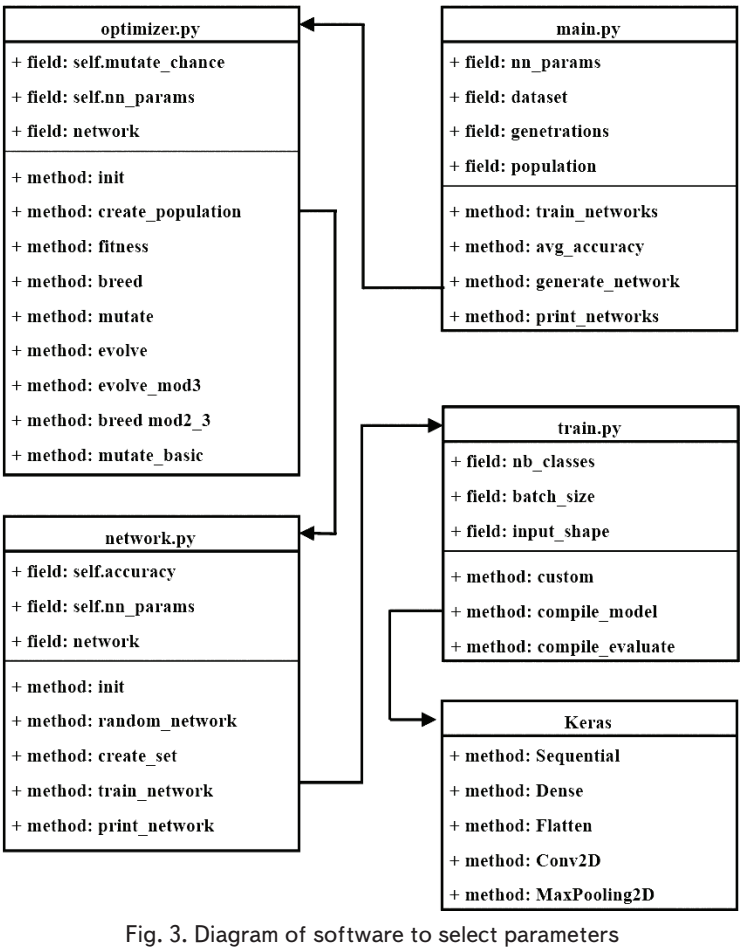

Fig. 3. Diagram of software to select parameters for training neural networks

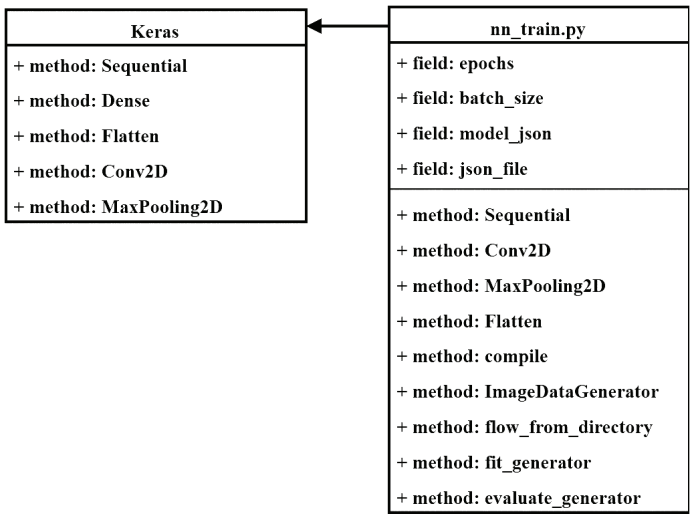

Fig. 4. Diagram of software to train a neural network

The neural network training software consists of file nn_train.py. This script downloads a training sample, trains a neural network, and stores it to a PC memory. The architecture of the trained neural network is sored to the file model.json, and the weights of the network are recorded to the file model.h5. The script uses the following methods:

– compile – constructs a neural network;

– ImageDataGenerator – generates a training set from the training sample;

– flow_from_directory – converts images for further use;

– fit_generator – works with images in the learning process;

– evaluate_generator – works with images in the process of auto-testing of recognition;

– save_weights – records weights of a neural network to the file model.h5.

The software of pattern recognition using a neural network consists of file nn_test. py that employs files from the network architecture model.json and the file of its weights model.h5. This script performs a readout of the input image, converts it to the appropriate format and submits to a neural network for recognition. The script uses the following methods:

– model_from_json – downloads the architecture of a neural network from file model.json;

– load_weights – downloads weights of a neural network from file model.h5;

– compile – constructs a neural network;

– load_img – downloads an image for recognition;

– predict – performs recognition based on an input image.

In the process of training the neural networks, we applied certain samples, which were used for training and testing recognition models by the developed modifications of simple genetic method. Each sample contains 5 classes: a pedestrian, a bicycle, a motorcycle, a car, a truck. All samples represent sets of images that contain the appropriate object. Each training sample consists of 1 million images, each testing sample – 400,000.

6. Experiments and results of studying the use of modifications of a simple genetic algorithm

The influence of values for parameters when training neural networks on an accuracy indicator and the time of learning was examined using a computer with the processor Intel Core i5 7400 with a clock frequency of 3 GHz, 8 GB of RAM.

The training of neural networks involved a learning sample, which consisted of 1 million images, divided into 5 classes.

We conducted 4 experiments in which we used the developed modifications of a genetic algorithm and a simple genetic algorithm itself. All experiments were carried out for the same parameters: the number of generations – 5, population size – 20, the mutation factor is 0.01. Input data – a sample of images from 5 classes (a pedestrian, a bicycle, a motorcycle, a car, a truck).

Results from experiments that compared a simple genetic method and the developed modifications are given in Table 1.

Based on the results given in Table 1 we can conclude that the use of the modification Alpha-Beta of a genetic algorithm is the best approach to achieve high recognition accuracy over less time.

Table 2 gives results of pattern recognition when using different approaches to train recognition models.

Based on Table 5, it can be concluded that a decrease in the number of classes for training from 100 to 5 has significantly increased the accuracy indicator, however, the selection duration has slightly increased. In addition, by comparing the results obtained during selection of parameters using a simple genetic algorithm and its modifications, we can conclude that the best approach to optimizing the process of selection of parameters for training a neural network is the use of the modification Alpha-Beta of a genetic algorithm.

Table 1

Results of parameters selection using a simple genetic algorithm

Version of GA	Learning duration		Accuracy on test data	Auto-testing accuracy
	Actual	Percentage to the GA basic version		
Basic	8 days, 8 minutes, 39 seconds	100 %	89.45 %	93.89 %
Modification Alpha-Beta	5 days, 22 hours, 27 minutes, 5 seconds	73.9 %	92.12 %	96.90 %
Modification Alpha-Beta fixed	8 days, 12 hours, 4 minutes, 58 seconds	106.2 %	90.02 %	95.89 %
Modification Fixed	7 days, 7 hours, 16 minutes,43 seconds	91.1 %	92.48 %	95.72 %

Table 2

Comparison of pattern recognition results using different approaches for training recognition models

Approach \ Characteristic	Learning duration	Accuracy on test data	Auto-testing accuracy
Cascade classifier	8 days, 15 hours, 36 minutes	90.95 %	–
Fully connected neural network (100 classes, no using GA)	1 hour, 4 minutes, 19 seconds	1.2 %	24.93 %
Convolutional neural network (100 classes, no using GA)	1 hour, 59 minutes, 34 seconds	1.8 %	47.14 %
Fully connected neural network (100 classes, using GA)	22 hours, 47 minutes, 33 seconds	1.4 %	30.52 %
Convolutional neural network (100 classes, using GA, 4 layers)	4 days, 16 hours, 59 minutes, 21 seconds	2.1 %	48 %
Convolutional neural network (100 classes, using GA, 6 layers)	6 days, 12 hours, 51 minutes, 42 seconds	2.39 %	53.69 %
Convolutional neural network (5 classes, using GA, 3 layers)	8 days, 8 minutes, 39 seconds	89.45 %	93.89 %
Convolutional neural network (5 classes, using the modification Alpha-Beta of GA, 3 layers)	5 days, 22 hours, 27 minutes, 5 seconds	92.12 %	96.90 %
Convolutional neural network (5 classes, using the modification Alpha-Beta fixed of GA, 3 layers)	8 days, 12 hours, 4 minutes, 58 seconds	90.02 %	95.89 %
Convolutional neural network (5 classes, using the modification Fixed of GA, 3 layers)	7 days, 7 hours, 16 minutes,43 seconds	92.48 %	95.72 %

7. Discussion of results of studying the use of modifications of a simple genetic algorithm

The result of this study is the developed modification of a simple genetic algorithm – Alpha-Beta. Using this modification makes it possible to speed up the implementation of selection of parameters to train neural networks, and to improve the accuracy indicator. It was proven that increasing the number of mutations and selection of different individuals to a pair ensured a greater variety of gene combinations, which leads to better indicators over less time.

Tables 1, 2 show that the developed modifications of a simple genetic method make it possible to increase the speed of synthesis of recognition models relative to the basic version of GA (specifically, using the modification Alpha-Beta requires only 73.9 % of the time required by a basic method, using the modification Fixed – 91.1 % of the time required by a basic genetic method). That made it possible to reduce the time for processing experimental array of data using the modified Alpha-Beta algorithm by 58 hours in comparison with a basic method. As can be seen from Table 2, the best method in terms of the rate of solving the selected problem is a fully connected neural network, which has the following characteristics: 1 hour, 4 minutes, 19 seconds; however, its accuracy of recognition is only 24.93 %. The fully connected neural network using GA has the following characteristics: learning time is 22 hours, 47 minutes, 33 seconds while the recognition accuracy is only 30.52 %. The developed modifications of a simple genetic method have made it possible to obtain the recognition accuracy of 95–96 %, which is an acceptable result.

Such results are predetermined by using the developed heuristic procedures, specifically mutations applying the Monte Carlo method, the modified selection and crossbreeding operators. This made it possible to improve the accuracy indicator and reduce the time of optimization using the developed modifications of a simple genetic method compared with its basic version.

Based on the recognition results, one can conclude that the trained neural network can determine that an individual belongs to a certain class at a very high accuracy.

Thus, the developed modifications of a simple genetic algorithm make it possible to improve recognition accuracy and reduce learning duration. This is achieved by using the new heuristic procedures in the developed modified methods, specifically we added a possibility of mutations using the Monte Carlo method, modified the operators of selection and crossbreeding. This has made it possible to improve the accuracy indicator compared to the basic version of a simple genetic algorithm.

The disadvantage of modifications of a simple genetic algorithm, developed and investigated in this work, is the need to spend much time (several days) when processing large arrays of data, which is not acceptable while resolving certain practical tasks. Thus, a limitation on using the developed modifications is a small amount of processed data.

Further advancement of this study could be associated with the elimination of these shortcomings, predetermined by a practical threshold in using the developed modifications. To this end, it is advisable to develop their parallel implementation, thereby significantly (by times) increasing the performance speed of methods. Related problems that could arise when developing parallel modifications of a simple genetic method, associated with the need to plan the resources for a parallel computer system and the increased requirements to the hardware employed in the process of genetic optimization.

8. Conclusions

1. We have developed three modifications of a simple genetic algorithm for the selection of parameters for training neural networks – Alpha-Beta, Alpha-Beta fixed, Fixed. In the developed modifications, in contrast to a simple genetic algorithm, we used the proposed heuristic procedures, such

as mutation using the Monte Carlo method, and modified operators of selection and crossbreeding. This has made it possible to improve the accuracy indicator and reduce the time of optimization by using the developed modifications of a simple genetic method compared with its basic version.

2. The software that has been implemented employs the developed modifications of a simple genetic algorithm for the selection of parameters for training neural networks and recognition of road users. When applying the developed modifications, the following indicators were obtained:

– Alpha-Beta: accuracy – 96.90 %, selection duration – 5 days, 22 hours, 27 minutes, 5 seconds;

– Alpha-Beta fixed: accuracy – 95.89 %, selection duration – 8 days, 12 hours, 4 minutes, 58 seconds;

– Fixed: accuracy – 85.48 %, selection duration – 7 days, 7 hours, 16 minutes, 43 seconds.

3. We have compared indicators for the modifications of a simple genetic algorithm and determined that the modification Alpha-Beta is the best approach to solve the task on recognition of road users; this modification enabled obtaining the best accuracy indicator over a shorter selection time.

Acknowledgement

The work was performed within the framework of the research theme «Methods and means of decision-making for data processing by the intelligent systems of pattern recognition» (State registration number 0117U003920) at the Department of Software at Zaporizhzhya National Technical University.

References

1. Dorosinskiy L. G. Raspoznavanie izobrazheniy, neyronnye seti i geneticheskie algoritmy // Advances in current natural sciences. 2011. Issue 10. P. 87–88.
2. Ekspert robototekhniki: «Nikogda ne ispol'zuyte avtopilot Tesla ryadom s velosipedistami!». URL: https://itc.ua/news/ekspert-robototehniki-nikogda-ne-ispolzuyte-avtopilot-tesla-ryadom-s-velosipedistami/
3. Mazda I-Activsense. URL: http://mazda.ua/ru/showroom/cx-5/i-activsense/
4. Waymo. URL: https://waymo.com/
5. Zhang Y., Ling Q. Bicycle Detection Based On Multi-feature and Multi-frame Fusion in Low-Resolution Traffic Videos // arXiv. 2017. URL: https://arxiv.org/pdf/1706.03309.pdf
6. Dalal N., Triggs B. Histograms of Oriented Gradients for Human Detection // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. doi: https://doi.org/10.1109/cvpr.2005.177
7. Encyclopedia of artificial intelligence / J. R. R. Dopico, J. Dorado, A. Pazos (Eds.). IGI Global, 2009. doi: https://doi.org/10.4018/978-1-59904-849-9
8. Support vector clustering / Ben-Hur A., Horn D., Siegelmann H. T., Vapnik V. // Journal of Machine Learning Research. 2001. Vol. 2. P. 125–137.
9. Viola P., Jones M. Rapid object detection using a boosted cascade of simple features // Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. 2001. doi: https://doi.org/10.1109/cvpr.2001.990517
10. Viola P., Jones M. J. Robust Real-Time Face Detection // International Journal of Computer Vision. 2004. Vol. 57, Issue 2. P. 137–154. doi: https://doi.org/10.1023/b:visi.0000013087.49260.fb
11. Urban traffic flow analysis based on deep learning car detection from CCTV image series / Peppa M. V., Bell D., Komar T., Xiao W. // ISPRS – International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2018. Vol. XLII-4. P. 499–506. doi: https://doi.org/10.5194/isprs-archives-xlii-4-499-2018
12. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks / Ren S., He K., Girshick R., Sun J. // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 39, Issue 6. P. 1137–1149. doi: https://doi.org/10.1109/tpami.2016.2577031
13. SSD: Single Shot MultiBox Detector / Liu W., Anguelov D., Erhan D., Szegedy C., Reed S., Fu C.-Y., Berg A. C. // Computer Vision – ECCV 2016. 2016. P. 21–37. doi: https://doi.org/10.1007/978-3-319-46448-0_2
14. Xuze Z., Shengsuo N., Teng H. A CNN Vehicle Recognition Algorithm based on Reinforcement Learning Error and Error-prone Samples // IOP Conference Series: Earth and Environmental Science. 2018. Vol. 153. P. 032052. doi: https://doi.org/10.1088/1755-1315/153/3/032052
15. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks // In NIPS. 2012. P. 1097–1105.
16. Leibe B., Seemann E., Schiele B. Pedestrian Detection in Crowded Scenes // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. doi: https://doi.org/10.1109/cvpr.2005.272
17. Fused Deep Neural Networks for Efficient Pedestrian Detection / Du X., El-Khamy M., Morariu V. I., Lee J., Davis L. // arXiv. 2018. URL: https://arxiv.org/pdf/1805.08688.pdf
18. Pedestrian Detection with Semantic Regions of Interest / He M., Luo H., Chang Z., Hui B. // Sensors. 2017. Vol. 17, Issue 11. P. 2699. doi: https://doi.org/10.3390/s17112699
19. Adu-Gyamf Y. O. Automated Vehicle Recognition with Deep Convolutional Neural Networks // Iowa State University Digital Repository. 2017. 12 p. URL: https://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=1182&context=ccee_pubs

20. Lipton A. J., Fujiyoshi H., Patil R. S. Moving target classification and tracking from real-time video // Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV'98 (Cat. No.98EX201). 1998. doi: https://doi.org/10.1109/acv.1998.732851
21. Cohen I., Medioni G. Detecting and tracking moving objects for video surveillance // Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149). 1999. doi: https://doi.org/10.1109/cvpr.1999.784651
22. Switchable Deep Network for Pedestrian Detection / Luo P., Tian Y., Wang X., Tang X. // 2014 IEEE Conference on Computer Vision and Pattern Recognition. 2014. doi: https://doi.org/10.1109/cvpr.2014.120
23. Tombari F., Salti S., Di Stefano L. Performance Evaluation of 3D Keypoint Detectors // International Journal of Computer Vision. 2013. Vol. 102, Issue 1-3. P. 198–220. doi: https://doi.org/10.1007/s11263-012-0545-4
24. Human Tracking Using Convolutional Neural Networks / Fan J., Xu W., Wu Y., Gong Y. // IEEE Transactions on Neural Networks. 2010. Vol. 21, Issue 10. P. 1610–1623. doi: https://doi.org/10.1109/tnn.2010.2066286
25. What is the best multi-stage architecture for object recognition? / Jarrett K., Kavukcuoglu K., Ranzato M. A., LeCun Y. // 2009 IEEE 12th International Conference on Computer Vision. 2009. doi: https://doi.org/10.1109/iccv.2009.5459469
26. Development of the indicator set of the features informativeness estimation for recognition and diagnostic model synthesis / Oliinyk A., Subbotin S., Lovkin V., Leoshchenko S., Zaiko T. // 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET). 2018. doi: https://doi.org/10.1109/tcset.2018.8336342
27. Oliinyk A. A., Subbotin S. A. A stochastic approach for association rule extraction // Pattern Recognition and Image Analysis. 2016. Vol. 26, Issue 2. P. 419–426. doi: https://doi.org/10.1134/s1054661816020139
28. Oliinyk A. O., Zayko T. A., Subbotin S. O. Synthesis of Neuro-Fuzzy Networks on the Basis of Association Rules // Cybernetics and Systems Analysis. 2014. Vol. 50, Issue 3. P. 348–357. doi: https://doi.org/10.1007/s10559-014-9623-7
29. Development of the method for decomposition of superpositions of unknown pulsed signals using the secondorder adaptive spectral analysis / Stepanenko A., Oliinyk A., Deineha L., Zaiko T. // Eastern-European Journal of Enterprise Technologies. 2018. Vol. 2, Issue 9 (92). P. 48–54. doi: https://doi.org/10.15587/1729-4061.2018.126578
30. Stratified model of the internet of things infrastructure / Alsayaydeh J. A. J., Shkarupylo V., Bin Hamid M. S., Skrupsky S., Oliinyk A. // Journal of Engineering and Applied Sciences. 2018. Vol. 13, Issue 20. P. 8634–8638.
31. Development of stratified approach to software defined networks simulation / Shkarupylo V., Skrupsky S., Oliinyk A., Kolpakova T. // Eastern-European Journal of Enterprise Technologies. 2017. Vol. 5, Issue 9 (89). P. 67–73. doi: https://doi.org/10.15587/1729-4061.2017.110142
32. Kolpakova T., Oliinyk A., Lovkin V. Improved method of group decision making in expert systems based on competitive agents selection // 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). 2017. doi: https://doi.org/10.1109/ukrcon.2017.8100388
33. Evolutionary Method for Solving the Traveling Salesman Problem / Oliinyk A., Fedorchenko I., Stepanenko A., Rud M., Goncharenko D. // 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T). 2018. doi: https://doi.org/10.1109/infocommst.2018.8632033