UDC 004.658

USING MODULAR NEURAL NETWORKS AND MACHINE LEARNING WITH REINFORCEMENT LEARNING TO SOLVE CLASSIFICATION PROBLEMS

Leoshchenko S. D. – PhD, Senior Lecturer of the Department of Software Tools, National University "Zaporizhzhia Polytechnic", Zaporizhzhia, Ukraine.

Oliinyk A. O. – Dr. Sc., Professor, Professor of the Department of Software Tools, National University "Zaporizhzhia Polytechnic", Zaporizhzhia, Ukraine.

Subbotin S. A. – Dr. Sc., Professor, Head of the Department of Software Tools, National University "Zaporizhzhia Polytechnic", Zaporizhzhia, Ukraine.

Kolpakova T. O. – PhD, Associate Professor, Associate Professor of the Department of Software Tools, National University "Zaporizhzhia Polytechnic", Zaporizhzhia, Ukraine.

ABSTRACT

Context. The solution of the classification problem (including graphical data) based on the use of modular neural networks and modified machine learning methods with reinforcement for the synthesis of neuromodels that are characterized by a high level of accuracy is considered. The object of research is the process of synthesizing modular neural networks based on machine learning methods with reinforcement.

Objective is to develop a method for synthesizing modular neural networks based on machine learning methods with reinforcement, for constructing high-precision neuromodels for solving classification problems.

Method. A method for synthesizing modular neural networks based on a reinforcement machine learning approach is proposed. At the beginning, after initializing a system of modular neural networks built on the bottom-up principle, input data is provided – a training set of data from the sample and a hyperparameter to select the size of each module. The result of this method is a trained system of modular neural networks. The process starts with a single supergroup that contains all the categories of the data set. Then the network size is selected. The output matrix is softmax, similar to the trained network. After that, the average probability of softmax is used as a similarity indicator for group categories. If new child supergroups are formed, the module learns to classify between new supergroups. The training cycle of modular neural network modules is repeated until the training modules of all supergroups are completed. This method allows you to improve the accuracy of the resulting model.

Results. The developed method is implemented and investigated on the example of neuromodel synthesis based on a modular neural network for image classification, which can later be used as a model for technical diagnostics. Using the developed method significantly reduces the resource intensity of setting up hyperparameters.

Conclusions. The conducted experiments confirmed the operability of the proposed method of neuromodel synthesis for image classification and allow us to recommend it for use in practice in the synthesis of modular neural networks as a basis for classification models for further automation of tasks of technical diagnostics and image recognition using big data. Prospects for further research may lie in using the parallel capacities of GPU-based computing systems to organize directly modular neural networks based on them.

KEYWORDS: modular neural networks, image classification, synthesis, diagnostics, topology, artificial intelligence, reinforcement learning.

ABBREVIATIONS

ANN is an artificial neural network;
BP is a backpropagation method;
CNN is a convolutional neural network;
GAN is a generative adversarial network;
GPU is a graphics processing unit;
MNN is a modular neural network;
PCA is a principal component analysis;
VAE is a variational autoencoders;
ViT is a vision transformer.

NOMENCLATURE

ΔAD is a density of accuracy changes;
$DataSet$ is a data set;
E is a relative error;
$Error_{ANN}$ is a neuromodel error;
$F1\ score$ is a F1 score;
FN represents false-negative results: the number of samples that the model predicted as not belonging to a class, while they actually belong to that particular class;
FP represents false positive results: the number of samples that were predicted to belong to the class, while they do not belong to the class at all;
i, j is a reference to the number of layers in two network modules;
$Model$ is a synthesized neuromodel;
MS is a block network size;
n is a number of input features that characterize sample instances;
$Precision$ is a precision of work of neuromodel;
$Recall$ is a recall of work of neuromodel;
TP represents true positive results: the number of samples that were predicted to belong to the class and correctly belong to the class;

DOI 10.15588/1607-3274-2024-2-8

VA is a verification accuracy obtained during image classification;

x_n is an independent attribute of the sample instance.

INTRODUCTION

Artificial intelligence technologies such as machine learning, deep learning, and computer vision can help use automation to structure and organize almost all data [1].

Images, including images and video glasses, provide the bulk of global data [2]. To interpret and systematize this data, developers of modern computing systems and software turn to image classification based on artificial intelligence [2].

Image classification analyzes digital images (photos, drawings, scanned copies, etc.) using analytical models based on artificial intelligence, which can identify and identify a wide range of criteria-from the content of the image to the time of day [1, 2].

Image classification is the task of classifying and assigning labels to groups of images or vectors within an image based on certain criteria [3]. A label can be assigned based on one or more criteria.

Image classification is a serious resource-intensive task for comparing modern architectures and methodologies in the field of Computer Vision [1, 3].

In a single-attribute classification, each image has only one label or annotation, as the name implies. As a result, for each image that the model sees, it analyzes and classifies it according to only one criterion [4].

On the other hand, when classified by multiple attributes, images can have multiple labels, and some images contain all the labels that you use at the same time [3].

The whole process begins with preprocessing – data preparation [5, 6].

This step improves image data by eliminating unwanted deformations and improving certain key aspects of the image so that computer vision models can work with these advanced data. In fact, data is cleaned up and prepared for processing by the artificial intelligence model [7, 8].

Data cleaning, sometimes called preprocessing, is an important step in preparing data for training your model, as data inaccuracies lead to inaccuracies in the image classification model [1–4]. As the results of clearing the data, you can see what happened:

– removing duplicates: duplicating data frees up the learning process and may cause the model to add more weight to duplicated data unnecessarily. [7, 8];

– deleting irrelevant data: including irrelevant data will not help train the model to achieve the desired goal [7, 8];

– filtering unwanted plague data (information outliers): some data, although technically relevant, do not help in training the artificial intelligence model [7, 8]. Data that goes beyond the norm can distort your model's predictions, so it's best to just delete it [3];

– missing data detection: missing data can cause problems in the learning process-during data cleaning, missing data can be identified and updated accordingly [7, 8];

– correction of structural errors: most machine learning methods are not able to identify errors as a human would, which means that each piece of data must be precisely organized [7, 8].

Well-organized data guarantees greater success when it comes to training an image classification model or any artificial intelligence model [7, 8].

Object detection: determines the location of objects in a set of images. This is the process of determining the location of an object, which includes image segmentation and determining the location of the object [4].

Object recognition and training: marking arranged images.

Deep learning methods reveal laws in the image and characteristics that may be unique to a particular feature.

The model is trained on this data set and becomes more accurate in the future [7, 8].

After the data has been identified, you need to learn your artificial intelligence model [7, 8]. This involves uploading a large amount of data to each of the attributes to give the artificial intelligence model the ability to learn something. The more training data is loaded, the more accurately the model will determine the content of each image [6].

Object classification: the model is ready to classify images. This is the final step in the development of an artificial intelligence model that classifies images according to several different criteria [7, 8].

The method uses an appropriate classification approach to classify the observed elements into predefined classes [1].

This is done by comparing the sample patterns with the desired patterns.

Now elements that were added as tags in the next step will be recognized by the algorithm in real images [7, 8].

However, most ANN training methods are still the same modifications of the BP method with certain adaptations to the more complex ANN architecture [11]. Moreover, such methods do not in any way touch the synthesis stage of the network topology. That is why the task of developing new methods based on the evolutionary approach for full-fledged ANN synthesis is urgent.

In modern computer vision, the model architecture is most often used as the most optimal standard by CNN [9].

The convolution operation introduces a matrix called a kernel into the input, performing pairwise multiplication of the Matrix and summing the result. Note that the filter and kernel are the same thing [10].

To apply the filter, imagine that it is applied to the input matrix, starting from the upper-right edge, multiply each value by the corresponding kernel value and sum them [9, 10]. Then the core is shifted by a certain amount, which is called a step, usually by one, and the process is repeated.

To apply the filter, imagine that it is applied to the input matrix, starting from the upper-right edge, each value

DOI 10.15588/1607-3274-2024-2-8

is multiplied by the corresponding kernel value and summed [9, 10]. Then the core is shifted by a certain amount, which is called a step, in this case by one, and the process is repeated again [9, 10].

The number of cores used simultaneously is called functions. For example, if you apply four cores 3×3 to an image, the resulting matrix will have four elements [9, 10].

In practice, core numbers are usually randomly initiated, put together, and updated during training using gradient descent [9, 10].

One of the key aspects of the convolution operation is the distribution of weights: most often, the same core, the gray mat, is reused at each step, covering the entire image [9, 10]. This significantly reduces the number of weights required, reducing calculation costs. Moreover, this leads to a shift in position, as the network learns to rely on the input matrix, combining pixels closely with each other. This helps because adjacent pixels are usually semantically related [9, 10].

Typically, in neural networks, several convolutional operation blocks are stacked together to form a layer. Several layers are then stacked together like constructors to form the final model. The number of layers is called depth; the more layers, the deeper the network [9, 10].

To further reduce the overhead of counting, different types of pools are applied between layers. Pool management is a complex process. It takes the Matrix and reduces it, summing up its values. The most common is the maximum Union, where the researcher determines the size of the window and takes the maximum value inside it [9, 10].

The final vector obtained by feeding an image packet to this series of convolution layers is then passed to a fully connected layer, which produces a vector with the classes [9, 10].

We present a comparison of the methods discussed above based on artificial neural networks in the following Table 1. we will use the following comparison criteria:

– structural complexity: the structural complexity of the network under consideration;

– using the GPU during training: it is possible to use the GPU during training in order to speed up this process;

– complexity of activation functions: assessment of the complexity and resource intensity of the activation function;

– working on large data sets: general assessment of the network's performance with large data samples.

Studding at the results of the comparison, we can conclude that today the general trend is the use of various modifications of CNN. At the same time, a not always more complex topology guarantees greater accuracy, as in the case of comparing the results of AlexNet and ResNet, because ResNet has a simpler and clearer architecture, but demonstrates higher accuracy. On the other hand, the use of a combination of layers of different topologies of artificial neural networks, as in the case of ViT, demonstrates the highest accuracy, but this imposes significant limitations in the use of modern computing capabilities (performing parallel calculations on the GPU). This is due to the fact that when performing calculations on parallel threads, constant synchronization and data forwarding between threads sometimes require significantly more resources than parallel calculations themselves.

Table 1 –Comparison of neural network models for image classification

	AlexNet	ResNet	ViT
Structural complexity	CNN-based, more simplified architecture (fewer layers, but more neurons)	CNN-based, more simplified architecture (fewer layers)	More complex, composite structure (layer Norm + MSP+ MLP)
Using the GPU during training	Possible	Possible	Possible (complicated by data forwarding)
Complexity of activation functions	Advanced function for improved accuracy (ReLU)	Advanced function for improved accuracy (ReLU)	Composite function for different blocks (GELU + softmax layers)
Working on large data sets	Possible (average accuracy and speed of operation)	Possible (medium accuracy and high speed of operation)	Possible (high accuracy and low speed)

That is why the task of finding the most optimal neuromodel architecture for image classification remains relevant, which would be characterized by topological simplicity and the possibility of full use of the capabilities of parallel calculation systems.

The object of study the process of MNN synthesis, based on reinforcement learning.

Existing solutions based on artificial neural networks are highly accurate, but they are characterized by excessive topological complexity and lack of the ability to fully use the capabilities of parallel computing systems.

The subject of the study MNN synthesis method using a machine learning approach with reinforcement learning.

Today, the general trend is the use of various modifications of deep ANNs. However, a more complex topology does not always guarantee greater accuracy. On the other hand, the use of a combination of layers of different ANN topologies demonstrates the highest accuracy, but complicates their implementation. Therefore, the paper proposes an approach based on the use of machine learning with reinforcement for MNN training, which allows combining different topologies and simplifying their synthesis.

The purpose of the work is to development of a method for MNN synthesis based on machine learning methods with reinforcement, for constructing high-precision neuromodels for solving classification problems.

1 PROBLEM STATEMENT

Let us define a sample of data consisting of images that are characterized by a number of features: $\{x_1, x_2, ..., x_n\}$. Then, using a model based on an artificial neural network: $Model = \{ANN\}$, you need to pass the entire sample of data $DataSet = \{x_1, x_2, ..., x_n, y\}$ through it, determine the class of each of the images, determine the dependent feature $\{y\}$. The conditions for regulating the learning process are to determine the reward/penalty during training. $Model = \{ANN\}$ in order to obtain a solution characterized by the greatest accuracy of work: $Error_{ANN} \to 0$.

2 REVIEW OF THE LITERATURE

To teach a model to classify images, huge amounts of data are required [13]. There are most often two approaches to machine learning models:

– supervised learning;

– unsupervised learning.

The more common of the two is learning with a teacher. This is when the expert advisor provides the system with data samples that already have a class designation attribute (or rule responses). This teaches the model to recognize correlations and apply procedures to new data [13].

Learning without a teacher is not as common as learning with a teacher. Learning without a teacher is characterized by sloppy, raw data without human intervention. This approach does not use basic data [14].

However, learning without a teacher can lead to solving problems when a person is an expert who has not yet been identified [13].

Supervised machine learning is a type of machine learning in which a model is trained on a set of data with markers, that is, the model is provided with input – output pairs. The goal is for the model to examine the mapping between inputs and outputs so that it can make predictions or make decisions when new, previously unmarked data arrives.

Supervised learning usually has clear stages.

Data collection. The work begins with collecting a data set consisting of the values of input object attributes and the corresponding class labels. For example, signs may include data such as color, size, age, gender, clinical indicators, and so on.

Preprocessing of data. Once the tag is created, preprocessing usually takes place to clean up and convert the data to a format suitable for training. This may include processing missing values, scaling functions, or encoding categorical variables.

Data separation. The entire sample must be divided into two or more subsets: usually this is a time for training the model and, accordingly, a part for testing. The training set is used to train the model, while the test set is used to evaluate its performance on invisible data.

Selecting a model. In the future, the structure of Delhi and the specific method of training are selected. The choice of model depends on various factors, such as the nature of the data, the complexity of the task, and the available computing resources.

Training the model. The model is trained based on training data that processes input objects along with the corresponding labels. During training, the model adjusts its internal parameters to minimize the difference between its forecasts and actual labels.

Evaluating the model. After training, you need to evaluate the performance of the model using a set of tests. This gives you an idea of how well model summarizes new, invisible data. Common evaluation measures include accuracy, precision, responsiveness, F1 score, and so on.

Setting up hyperparameters. In many cases, you may need to fine-tune the model's hyperparameters to further improve its performance. Hyperparameters are config parameters that are not studied during training, but affect the learning process.

Deployment. Once a satisfactory level of model performance is reached, you can deploy it to make predictions based on new, unknown data in real-world applications.

Throughout this process, it is important to repeat and improve your approach based on the performance of the model and the conclusions obtained from the analysis of results. Supervised learning is widely used in various fields, including, but not limited to, regression, classification, and ranking tasks.

Unsupervised machine learning is a type of machine learning in which you have input data without the appropriate labels. The goal is to identify hidden laws, structures, or relationships within them. Unlike learning with a teacher, there are no pre-determined results or correct answers that guide the learning process. The method attempts to find internal structures or representations in the data.

Unsupervised learning is usually similar in stages to learning with teachers.

Data collection. As with learning with a teacher, the robot starts by collecting a set of data. However, when teaching without a teacher, the data set consists only of input objects without any corresponding labels or target variables.

Preprocessing of data. Preprocessing is similar to supervised learning, processing missing values, scaling functions, encoding categorical variables, and so on.

Selecting a model. Choose a teaching method without a teacher or a model that matches the task. Advanced unsupervised learning methods include clustering, dimensionality reduction, and generative modeling.

Training the model. When teaching without a teacher, Model learns exclusively from input data without explicit control. The method examines the structure of data and examines patterns or representations that reflect their main characteristics.

Clustering. Clustering methods group similar data points based on some degree of similarity. K-means clustering and hierarchical clustering are examples of popular clustering methods.

DOI 10.15588/1607-3274-2024-2-8

Dimensionality reduction: dimensionality reduction methods are aimed at reducing the number of objects while maintaining as much relevant information as possible. PCA and t-SNE are commonly used to reduce dimensionality.

Generative modeling: generative models learn the basic probability distribution of data and can generate new patterns similar to training data. Examples are autoencoders, VAE, and GAN.

Evaluating the model. Unlike learning with a teacher, there may not always be clear assessment indicators for learning tasks without a teacher. Evaluation often involves qualitative evaluation by checking learned representations, visualizing clusters, or evaluating the quality of non-selected samples.

Interpretation and application: once the model is trained, it is possible to interpret the learned patterns or representations and apply them to solve specific problems or obtain information about the data.

Unsupervised learning is used in a variety of applications, such as customer segmentation, anomaly detection, data compression, and character learning. This can be especially useful when tagging data is scarce or expensive, or when learning and understanding the data structure is the main goal.

Reinforcement learning is an area of machine learning that considers taking appropriate steps to maximize reward in a particular situation [13–16]. This approach is used by various software and machines to find the best possible behavior or path to choose in a particular situation. Reinforcement learning differs from training with a teacher in that when training with a teacher, the training yes-no contains a key to the answer, so the model learns with the correct answer by itself, whereas when training with reinforcement, there is no answer, but the reinforcement agent decides what to do to complete the task. In the absence of a training data set, he must learn from his own experience [13–16].

A well-known example shows a robot, a diamond, and a fire. The goal of the robot is to get a reward in the form of a diamond and avoid obstacles that are triggered [13–16]. The robot learns, tries all possible paths, and then chooses the one that gives it the reward with the fewest cross-codes. Each correct step will give the robot a reward, and each wrong step will take the reward from the robot. The total reward will be calculated when it reaches the final reward-a diamond. [13–16]

Basic principles of reinforcement learning [13–16]:

– input data: the input data must be the initial state from which the model will start;

– result: there are many possible outcomes, as there are many solutions to a particular problem;

– training is based on input data, the model will return the state, and the user will decide to reward or punish the model based on its results;

– the model continues to learn.

– the best decision is made on the basis of a mock reward.

Let's compare machine learning methods for classifying images by the following criteria:

– input requirements: overall assessment of the machine learning approach and appropriate methods for organizing and presenting input data;

– requirements for training metaparameters: requirements for parameters and settings necessary for using the training process;

– complexity of the organization: assessment of the complexity of organizing the model training process, including in terms of computing resources;

– adaptability of solutions (models): assessment of the adaptability and universality of the obtained Solutions, their ability to work with updated data about the task (or environment).

The results of the analysis are presented in the form of a Table 2.

Table 2 – Comparison of machine learning approaches

	Supervising learning	Unsupervised learning	Reinforcement learning
Input requirements	The highest	Minimum	Average
Requirements for training weather parameters	High	High	Average
Complexity of the organization	Average	High	High
Adaptability of solutions (models)	Average	High	The highest

From the results of the comparison, it can be concluded that in general, learning with and without a teacher, as the main approaches to machine learning, already take into account a large number of developed and studied advantages and disadvantages. Thus, the very concept of learning with a teacher requires a detailed preprocessing of input data with their clear classification in the educational part. Learning without a teacher does not require this from the data and can even work with them without much verification, but most methods of learning without a teacher require prior determination of many metaparameters to start the learning process.

Reinforcement learning is a more modern approach that attempts to solve the known problems of previous strategies. Yes, it has requirements for input data about the environment, but they are not excessive, and metaparameters are usually limited to defining criteria for stopping and determining the success/failure of training. Also, Solutions (mods) obtained on the basis of reinforcement learning are most adaptable and versatile, because they are immediately synthesized taking into account the variability of the environment in which the agent operates.

3 MATERIALS AND METHODS

The main idea of MNN is based on modularity, and is that: the best approach to solving a given complex problem is not to perform one giant task, but a system of separate and mostly independent subtasks working together to

DOI 10.15588/1607-3274-2024-2-8

achieve one big goal [17–20]. This concept has a biological basis. There are functionally specialized areas in the brain that are specific to various cognitive processes. In a part of the brain called the thalamus, there is a lateral cranial nucleus, which is divided into layers that separately process color and color: both the main components of vision [17–20]. MNN use this idea to solve complex artificial intelligence problems. Several independent neural networks are simultaneously trained for a specific subtask, and their results are eventually combined to perform a single task. Advantages of MNN include [17–20]:

– simplicity of structural structure and organization;
– a combination of teaching techniques and methods;
– scalability of solutions;
– efficient use of computing systems.

As you know, artificial intelligence does not inherently speak language, so the main task is to teach an analytical model to reason over a given text. Reasoning refers to tasks such as arithmetic, sorting, comparison, and counting [17–20].

For example, you need to create a system of responses to questions Fig. 1 illustrates how the quality control model works [17–20].

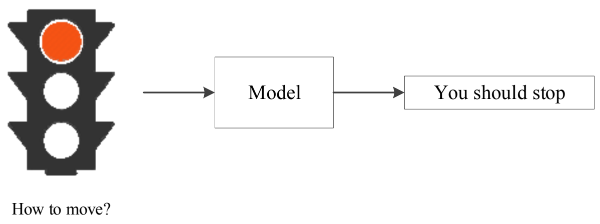

Figure 1 – Visualization of the answer to a question-the task is to answer a question about an image to show that the system understands the image

So, in the problem with determining what the sign indicates: the color mark of the traffic light in Fig. 1. To answer this question, you need to perform several reasoning steps: find a traffic light, determine the color mark, find out the color, and then provide a decision on further actions [17–20].

MnMs are able to provide such reasoning. Using this question, the model builds a specific network architecture, and then performs the assembled sequence of neural modules to output the answer, as shown in Fig. 2 [17–20].

MNNs refer to artificial neural networks that consist of several different neural networks connected together in combination with an intermediary. To illustrate this point further, consider a consumer who owns several smart devices, such as a smartphone, smartwatch, and tablet, such as an iPad, in addition to a laptop or desktop computer [17–20]. Despite the different capabilities of these respective devices, all of them will be connected to a modem or traffic jam route, which will allow users of these devices to quickly and efficiently access online and mobile services. In addition, this online connection also allows users to combine the functionality of their various devices to achieve a specific goal, such as streaming a popular television program or calling a friend or family member, among other things [17–20].

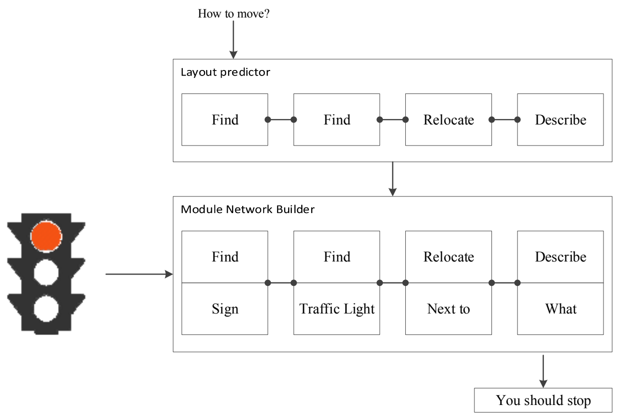

Figure 2 – For each instance, the model first assumes a layout, then, using the image functions, it builds the assembled network of neural modules to output the response

Taking all of the above into account, MNNs enable software developers to use the capabilities of individual neural networks in a more consistent and efficient way. To do this, each neural network within a larger MNN will be used to solve part of a specific problem [17–20]. At this point, an intermediary known as an integrator will be used to organize and analyze these many modules to create the final result of the neural network. Thanks to this configuration, simple neural networks can be implemented in a more complex way, and some common applications of these networks include image processing, high-level input compression, and stock market forecasting software [17–20].

Ensemble training. The concepts and ideas that formed the basis for creating modular neural networks were first theorized in the 1980s and led to the development of a machine learning method called ensemble learning [17–20]. This method is based on the idea that weaker machine learning models can be combined together to create a single stronger Model [17–20]. Moreover, this collective approach can be used to produce more significant results than those that could be obtained using a single deep learning model. Another way to conceptualize this process is the divide-and-conquer approach, in which a large problem is solved by breaking it down into smaller parts that can be solved in a simpler or more viable way [17–20].

MNN structure. Consistent with the idea that neural networks are based on multiple functions and capabilities of the human brain, the human brain consists of a hierarchy of networks consisting of millions of neurons, with each network specialized to perform specific work related to the functioning of the human body [17–20]. To give an example of this, let's look at 3 different direct-link neural networks that have been trained to solve problems related to pattern recognition, such as license plate recognition. For many reasons, these 3 networks may have difficulty analyzing and identifying license plates in videos or images on an ongoing basis [17–20]. Thus, the software developer could combine the source data of these three models to create a single output signal in order to create a

DOI 10.15588/1607-3274-2024-2-8

single model that is more accurate than the previous 3 models [17–20].

In addition, software developers can also use MNN to break down the learning problem itself into smaller, more manageable parts [17–20]. Going back to the example of license plate recognition, this problem can be divided into three subtasks, such as identifying license plates in images, identifying license plates in videos, and identifying license plates in images that depict bad weather, such as heavy rain or snowfall. Thanks to these combined conclusions, the developer could create a deep learning algorithm that would be able to identify the appearance of license plates in both images and videos depicting various weather conditions that a driver may encounter when passing a busy intersection or street [17–20].

Using MNN offers a wide range of benefits. For example, combining multiple neural networks that have been trained to perform a specific task can significantly reduce the training time that is often required to train neural networks, as well as the associated costs [17–20]. Conversely, this approach also allows software developers to combine different types of machine learning models and approaches that would not be possible with traditional methods. Finally, as already mentioned, the combination of several neural networks can lead to the development of a single neural network that can work much more accurately and efficiently than any of the previous networks were capable of independently [17–20].

Since artificial neural networks have generally become a hot topic for both research and software development over the past few years, it is appropriate that MNN is an approach that is also becoming increasingly popular [17–20]. Moreover, since the cost of training machine learning models in practice can be exorbitant, MNNs allow software developers to create models much cheaper and more sustainable. At the same time, MNNs will continue to be used in the coming years, since approaches that were considered inadequate in previous scenarios can be combined to create a single viable solution [17–20].

In the usual formulation of the problem of image classification as a computer vision problem, it is reduced to maximizing the confidence function from a set of hypothetical target locations, where reliability can be studied in a fully controlled or poorly controlled setting [17–20]. In the formulation of a sliding window, the hypothesis set consists of a large set of rectangular windows, and the maximization problem is solved by exhaustive search. Since this process, as a rule, is too expensive in practice, many methods have been proposed to speed it up, from methods that use the properties of a confidence function, to sentence methods or cascade methods. All these methods retain the property of exhaustive search in the hypothesis space, aimed either at reducing the number of hypotheses to begin with, or at effectively searching for them [17–20].

In general, the advantages of using subcripping training in image classification are as follows:

– reinforcement learning can be used to solve very complex problems that cannot be solved by conventional methods when the input volumes are too large [17–20];

– this approach is preferred for achieving high rates of adaptability and versatility of solutions [17–20];

– provides an opportunity to correct mistakes made in the learning process [17–20];

– as soon as the error is corrected by the model, the probability of occurrence of the same error is very small;

– in the absence of a training data set, the sub-course must learn from its own experience [17–20];

– reinforcement learning models can translate people into many tasks [17–20];

– reinforcement learning is designed to achieve the ideal behavior of a model in a particular text in order to maximize its performance [17–20];

– reinforcement learning methods maintain a balance between research and exploitation. Research is the process of testing different things to see if they are better than what has been done before. Exploitation is the process of testing what has worked best in the past. Other learning algorithms do not provide such a balance [17–20].

To build an MNN system, you need to select the appropriate network size for each module, and then find the average softmax probability for playing categories. The MNN system is built on the bottom-up principle. The input data for the method is a training dataset from the sample and a hyperparameter for selecting the size of each module. The result of the method is a trained MNN system [17–20]. First, we start with a single supergroup that contains all the categories of the data set. Then the network size is determined. The output matrix is softmax, similar to the trained network. After that, the average probability of softmax is used as a similarity indicator for group categories. If new child supergroups are formed, the module is trained to classify between new supergroups. Training of MNN modules is described below. The cycle is repeated until the modules of all superg-Rup [17–20] are trained.

Selecting neural network sizes and hyperparames. You must select the number of layers and the size (configuration) of each module. You should use a new metric called the accuracy change density [17–20] (ΔAD). It measures the accuracy gain per unit increase in model size between two network models:

$$\Delta AD_{ij} = \frac{VA_i - VA_j}{MS_i - MS_j}. \tag{1}$$

If we consider the efficiency of use ΔAD as an indicator, it should be noted that when increasing the size of the model, there is no noticeable increase in accuracy. The accuracy is calculated for each MNN module before other networks are grouped into supergroups [17–20]. For example, ΔAD for a root module, MNN is calculated based on the accuracy obtained for classification between all categories of the data set, and for any other module in MNN, it is calculated by classification between all its child categories [17–20]. Using ΔAD, you can define effective network configurations and distinguish categories with high accuracy. This approach can lead to a slight loss of accuracy compared to large monolithic neural networks, since small networks with MNN need to classify only several groups of visually similar categories, rather than all categories of the data set.

Fig. 3 shows a common variant of communication between MNN modules during operation. This is how the main module is allocated, which is responsible for distributing operations and further synchronizing their results. Fig. 4 shows a schematic example of simulated models.

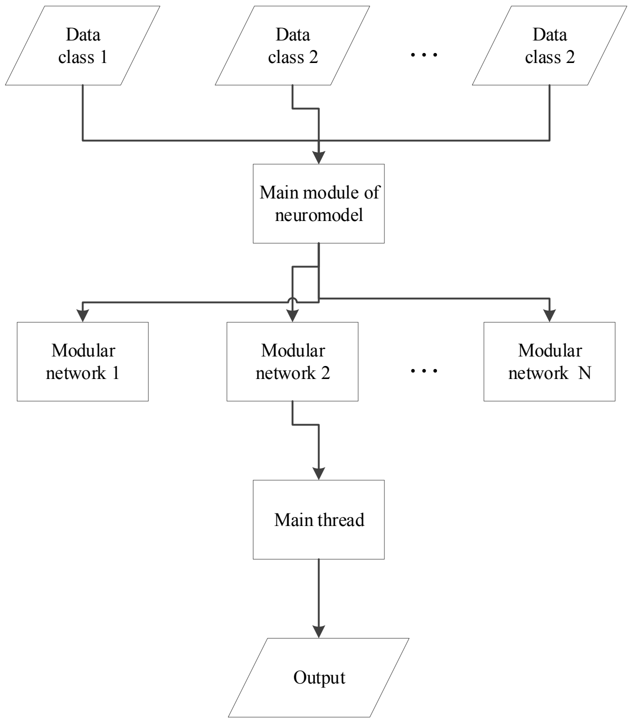

Figure 3 – General representation of communication between MNN modules during operation

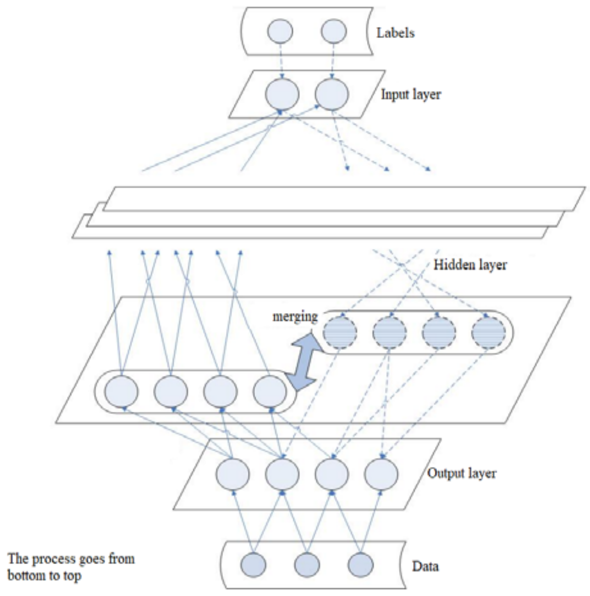

Figure 4 – Schematic example of synthesized networks

4 EXPERIMENTS

Several sets of image data were used in the experiments. CEFAR [27–29], SVHN [30], and EMNIST [31], which contain centered and fixed-size images with only one object in each image. Images in large data sets such as ImageNet 2012 [32] and Caltech-256 [33] have different sizes and display real images more accurately.

The CEFAR datasets [27–29] consist of color images of 32×32 different categories in size. The training and test kits contain 50 thousand and 10 thousand images, respectively. During the experiments, the generally accepted practice of using 5,000 images from the training set to form a test set was observed [27–29]. The SVHN dataset contains 73,257 pixel-sized color images in the training set 32×32 and 531,131 images for additional training [30]. When reporting the results of the SVHN data set, it was generally accepted to use all training data without any increase in data. A set of 6000 images is used to check learning outcomes. EMNIST is an extension of the popular MNIST dataset [31]. There are six configurations of the EMNIST dataset, and an EMNIST-balanced configuration was used. It contains 131,600 pixel – sized grayscale images belonging to 47 categories. The ImageNet training set contains 1000 categories of approximately 1000 images each [33] 28×28. ImageNet also includes a verification kit and a testing kit. A subset of the ImageNet dataset with 20 categories was also used to easily visualize MNN and fully understand the details and properties of the hierarchy. The Caltech dataset is also used in experiments [32]. As suggested in [32], a subset of 11 categories from the Caltech dataset was used to provide a fair comparison with existing work. Each category of the Caltech dataset contains approximately 100 training images and 20 test images. The parameters of the data sets are described in Table 3.

Table 3 – Detailed information about sampling data for testing

Data set	Size of the images	Number of images to study	Number of images to test	Number of image classes
CIFAR 10	$32\times32\times3$	50000	10000	10
CIFAR 100	$32\times32\times3$	50000.	10000	100
SVHN	$32\times32\times3$	604388	26032	10
EMNIST	$28\times28\times1$	112800	18800	47
ImageNet 2012	Different	1200000	75000	1000
ImageNet 2012	Different	26000	2000	20
Caltech	Different	2000	400	11

Image classification models should be evaluated to determine how well they perform compared to other models [7, 8]. Here are some well-known indicators used in image classification [7, 8].

Accuracy. Accuracy is an indicator that is defined for each class. Class accuracy tells us how much of the data provided by the machine learning model for class membership was actually part of the class in the validation data. A simple formula can demonstrate this [7, 8]:

DOI 10.15588/1607-3274-2024-2-8

$$Precision = \frac{TP}{TP + FP}. \qquad (2)$$

Completeness. Completeness, similar to accuracy, is defined for each class [7, 8].

Completeness tells us what proportion of data from the validation set belonging to the class was correctly defined (as belonging to the Class) [7, 8].

Completeness can be represented as [7, 8]:

$$Recall = \frac{TP}{TP + FN}. \qquad (3)$$

F1 Rating. The F1 score helps to achieve a balance between accuracy and completeness in order to get an average idea of how the Model Works [7, 8]. The F1 score as an indicator is calculated as follows:

$$F1\,Score = \frac{2 \cdot (Precision \times Recall)}{Precision + Recall}. \qquad (4)$$

Relative error. The relative error in this case will be calculated as the ratio of the classification error to the total sample size (the number of its instances).

$$E = \frac{error_{class}}{Number_{sampl}} \cdot 100\%. \qquad (5)$$

5 RESULTS

A comparison of MNN test results with other neural networks when classifying selected data samples is shown in Table 4.

Table 4 – General test results

Dataset	ANN	Model size in memory (KB)	Model error on test data
CIFAR-10	VGG-16	78.410	0.067
	VGG-Pruned	28.200	0.066
	MNN	806	0.079
CIFAR-100	VGG-16	78.590	0.295
	VGG-Pruned	28.910	0.252
	ResNet	141.100	0.192
	MNN	832	0.209
SVHN	ResNet	11.000	0.016
	MNN	522	0.018
	EDEN	—	0.117
EMNIST	**MNN**	363	0.078
	VGG-16	528.000	0.076
ImageNet 2012 (subset)	ResNet	84.000	0.081
	MNN	1.872	0.124
	VGG-16	528.120	0.295
ImageNet 2012	ResNet-34	84.100	0.276
	MNN	2.515	0.313

6 DISCUSSION

The MNN architecture was compared with several existing architectures.

MNN has been shown to have the lowest memory requirements and the number of iterations. This performance boost is achieved with a small test error cost. ResNet and VGG contain 54 and 16 layers, respectively. In some tests, it was also recommended to use a shortened and quantum version of the VGG architecture: VGG-Pruned. MNN also equaled ResNet. For the EMNIST dataset, we use EDEN for comparison.

These architectures contain large deep neural networks with inverted bottleneck filters to reduce the number of operations.

Table 4 shows that the MNN tree has the smallest model size. Compared to VGG-Pruned on the CIFAR – 100, MNN requires a 97.12% smaller model. Smaller models require fewer memory accesses, achieve faster convergence, and reduce power consumption. The table shows the number of floating-point multiplications and additions performed during the output of a single image. The specified model size and the number of MNN operations are the sum of module values along the longest run path.

There is a slight difference in memory requirements and number of operations when comparing the MNN for ImageNet 2012 (subset) and the entire ImageNet 2012 dataset. this shows the scalability of the proposed method when building using the methods presented in Section 1. the table does not report the model size and number of operations for EDEN because the data and source code are not publicly available.

Table 4 shows that MNN achieves the lowest error of 7.8% for the EMNIST dataset. The accuracy of MNN is comparable to the state of the art for the CIFAR-10 and SVHN datasets. It is worth noting that the test error achieved in modern monolithic deep neural network architectures is achieved after significant efforts to configure hyperparameters.

CONCLUSIONS

The actual scientific and applied problem of developing a method for synthesizing MNN based on machine learning methods with reinforcement, for constructing high-precision neuromodels for solving classification problems.

The scientific novelty lies in the fact that a method has been developed implemented and investigated on the example of neuromodel synthesis based on a modular neural network for image classification, which can later be used as a model for technical diagnostics.

The practical significance lies in the fact that the using the developed method significantly reduces the resource intensity of setting up hyperparameters.

Prospects for further research may lie in using the parallel capacities of GPU-based computing systems to organize directly modular neural networks based on them.

ACKNOWLEDGEMENTS

The work was carried out with the support of the state budget research project of the state budget of the National University "Zaporozhzhia Polytechnic" "Intelligent information processing methods and tools for decision-making in the military and civilian industries" (state registration number 0121U107499).

REFERENCES

1. Kuo C. Transfer Learning for Image Classification: With Python Examples. New York, Kindle, 2022, 127 p.
2. Borra S., Thanki R., Dey N. Satellite Image Analysis: Clustering and Classification (SpringerBriefs in Applied Sciences and Technology). Cham, Springer, 2019, 113 p.
3. Siahaan V. Step by Step Tutorial image classification Using Scikit-Learn, Keras, And TensorFlow with Python. New York, Kindle, 2021, 141 p.
4. Canty M. J. Image Analysis, Classification and Change Detection in Remote Sensing: With Algorithms for Python. New York, CRC Press, 2019, 530 p.
5. Image Classification [Electronic resource]. Mode of access: https://www.sciencedirect.com/topics/engineering/image-classification
6. Image Classification Explained [+V7 Tutorial] [Electronic resource]. Mode of access: https://www.v7labs.com/blog/image-classification-guide#h4
7. Image Classification in AI: How it works [Electronic resource]. Mode of access: https://levity.ai/blog/image-classification-in-ai-how-it-works
8. Image Classification [Electronic resource]. Mode of access: https://huggingface.co/tasks/image-classification
9. Dayhoff J. Neural Network Architectures: An Introduction. New York, Van Nostrand Reinhold Company, 1990, 259 p.
10. Aggarwal C.C. Neural Networks and Deep Learning: A Textbook. Cham, Springer, 2018, 520 p.
11. Rashid T. Make Your Own Neural Network. CreateSpace Independent Publishing Platform, 2016, 222 p.
12. Jones H. Neural Networks: An Essential Beginners Guide to Artificial Neural Networks and their Role in Machine Learning and Artificial Intelligence. CreateSpace Independent Publishing Platform, 2018, 76 p.
13. Theobald O. Machine Learning for Absolute Beginners: A Plain English Introduction (Machine Learning with Python for Beginners). London, Scatterplot Press, 2020, 181 p.
14. Sebastian C. Machine Learning for Beginners: Absolute Beginners Guide, Learn Machine Learning and Artificial Intelligence from Scratch. New York, Kindle, 2018, 102 p.
15. Julian D. Designing Machine Learning Systems with Python. Birmingham, Packt Publishing, 2016, 232 p.
16. Burkov A. The Hundred-Page Machine Learning Book. New York, Kindle, 2022, 160 p.
17. Modular Neural Networks for Low-Power Image Classification on Embedded Devices [Electronic resource]. Mode of access: https://dl.acm.org/doi/10.1145/3408062
18. Reasoning Using Modular Neural Networks [Electronic resource]. Mode of access: https://towardsdatascience.com/reasoning-using-modular-neural-networks-f003cb6109a2
19. What is a Modular Neural Network? Weak vs. Strong Learners [Electronic resource]. Mode of access: https://caseguard.com/articles/what-is-a-modular-neural-network-weak-vs-strong-learners/
20. Modular neural networks using multiple gradients [Electronic resource]. Mode of access: http://essay.utwente.nl/82225/
21. The CIFAR-10 dataset [Electronic resource]. Mode of access: https://www.cs.toronto.edu/~kriz/cifar.html
22. The CIFAR-100 dataset [Electronic resource]. Mode of access: https://www.cs.toronto.edu/~kriz/cifar.html
23. CIFAR-100 [Electronic resource]. Mode of access: https://paperswithcode.com/dataset/cifar-100
24. The Street View House Numbers (SVHN) Dataset [Electronic resource]. Mode of access: http://ufldl.stanford.edu/housenumbers/
25. The EMNIST Dataset [Electronic resource]. Mode of access: https://www.nist.gov/itl/products-and-services/emnist-dataset
26. ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) [Electronic resource]. Mode of access: https://www.image-net.org/challenges/LSVRC/2012/
27. Caltech 256 Image Dataset [Electronic resource]. Mode of access: https://www.kaggle.com/datasets/jessicali9530/caltech256

Received 18.03.2024.
Accepted 01.05.2024.

УДК 004.896

ВИКОРИСТАННЯ МОДУЛЯРНИХ НЕЙРОННИХ МЕРЕЖ ТА МАШИННОГО НАВЧАННЯ З ПІД-КРІПЛЕННЯМ ДЛЯ ВИРІШЕННЯ ЗАДАЧ КЛАСИФІКАЦІЇ

Леощенко С. Д. – д-р філософії, ст. викладач кафедри програмних засобів Національного університету «Запорізька політехніка», Запоріжжя Україна.

Олійник А. О. – д-р техн. наук, професор, професор кафедри програмних засобів Національного університету «Запорізька політехніка», Запоріжжя, Україна.

Субботін С. О. – д-р техн. наук, професор, завідувач кафедри програмних засобів Національного університету «Запорізька політехніка», Запоріжжя, Україна.

Колпакова Т.О. – канд. техн. наук, доцент кафедри програмних засобів Національного університету «Запорізька політехніка», Запоріжжя, Україна.

АНОТАЦІЯ

Актуальність. Розглянуто вирішення задачі класифікації (в тому числі графічних даних) на основі використання модулярних нейронних мереж та модифікованих методів машинного навчання з підкріпленням для синтезу нейромоделей, які відрізняються високим рівнем точності роботи. Об'єктом дослідження є процес синтезу модулярних нейронних мереж на основі методів машинного навчання з підкріпленням.

DOI 10.15588/1607-3274-2024-2-8

Мета роботи полягає у розробці методу синтезу модулярних нейронних мереж на основі методів машинного навчання з підкріпленням, для побудови нейромоделей високої точності для розв’язання задач класифікації.

Метод. Запропоновано метод синтезу імпульсних нейронних мереж на основі еволюційного підходу. На початку, після ініціалізації системи модулярних нейронних мереж, що побудована за принципом знизу вгору, подаються вхідні дані – навчальний набір даних з вибірки і гіперпараметр, для вибору розміру кожного модуля. Результатом роботи методу є навчена система модулярних нейронних мереж. Процес починають з однієї супергрупи, яка містить усі категорії набору даних. Потім обирається розмір мережі. Вихідна матриця softmax, подібна для навченої мережі. Після чого середня ймовірність softmax використовується як показник подібності для групових категорій. Якщо формуються нові дочірні супергрупи, модуль навчається класифікації між новими супергрупами. Цикл навчання модулів модулярних нейронних мереж повторюється до тих пір, поки не будуть навчені модулі всіх супергруп. Метод дозволяє підвищити точність результуючої моделі.

Результати. Розроблений метод реалізовано та досліджено на прикладі синтезу нейромоделі на основі модулярної нейронної мережі для класифікації зображень, яка в подальшому зможе використовуватися у якості моделі для технічного діагностування. Використання розробленого методу значно знижує ресурсоємність налаштування гіперпараметрів.

Висновки. Проведені експерименти підтвердили працездатність запропонованого методу синтезу нейромоделі для класифікації зображень та дозволяють рекомендувати його для використання на практиці при синтезі модулярних нейронних мереж у якості основи класифікаційних моделей для подальшої автоматизації задач технічного діагностування та розпізнавання образів з використанням великих даних. Перспективи подальших досліджень можуть полягати у використанні паралельних потужностей обчислювальних систем на базі GPU для організації на їх базі безпосередньо модульних нейромереж.

КЛЮЧОВІ СЛОВА: модулярні нейронні мережі, класифікація зображень, синтез, діагностування, топологія, штучний інтелект, навчання з підкріпленням.

ЛІТЕРАТУРА

1. Kuo C. Transfer Learning for Image Classification: With Python Examples / C. Kuo. – New York : Kindle, 2022. – 127 p.
2. Borra S. Satellite Image Analysis: Clustering and Classification (SpringerBriefs in Applied Sciences and Technology) / S. Borra, R. Thanki, N. Dey. – Cham : Springer, 2019. – 113 p.
3. Siahaan V. Step by Step Tutorial image classification Using Scikit-Learn, Keras, And TensorFlow with Python GUI / V. Siahaan. – New York : Kindle, 2021. – 141 p.
4. Canty M. J. Image Analysis, Classification and Change Detection in Remote Sensing: With Algorithms for Python / M. J. Canty. – – New York : CRC Press, 2019. – 530 p.
5. Image Classification [Electronic resource]. – Access mode: https://www.sciencedirect.com/topics/engineering/image-classification
6. Image Classification Explained [+V7 Tutorial] [Electronic resource]. – Access mode: https://www.v7labs.com/blog/image-classification-guide#h4
7. Image Classification in AI: How it works [Electronic resource]. – Access mode: https://levity.ai/blog/image-classification-in-ai-how-it-works
8. Image Classification [Electronic resource]. – Access mode: https://huggingface.co/tasks/image-classification
9. Dayhoff J. Neural Network Architectures: An Introduction / J. Dayhoff. – New York : Van Nostrand Reinhold Company, 1990. – 259 p.
10. Aggarwal C. C. Neural Networks and Deep Learning: A Textbook / C. C. Aggarwal. – Cham : Springer, 2018. – 520 p.
11. Rashid T. Make Your Own Neural Network / T. Rashid. – Scotts Valley : CreateSpace Independent Publishing Platform, 2016. – 222 p.
12. Jones H. Neural Networks: An Essential Beginners Guide to Artificial Neural Networks and their Role in Machine Learning and Artificial Intelligence / H. Jones. – Scotts Valley : CreateSpace Independent Publishing Platform, 2018. – 76 p.
13. Theobald O. Machine Learning for Absolute Beginners: A Plain English Introduction (Machine Learning with Python for Beginners) / O. Theobald. – London : Scatterplot Press, 2020. – 181 p.
14. Sebastian C. Machine Learning for Beginners: Absolute Beginners Guide, Learn Machine Learning and Artificial Intelligence from Scratch / C. Sebastian. – New York : Kindle, 2018. – 102 p.
15. Julian, D. Designing Machine Learning Systems with Python / D. Julian. – Birmingham : Packt Publishing, 2016. – 232 p.
16. Burkov, A. The Hundred-Page Machine Learning Book / A. Burkov. – New York : Kindle, 2022. – 160 p.
17. Modular Neural Networks for Low-Power Image Classification on Embedded Devices [Electronic resource]. – Access mode: https://dl.acm.org/doi/10.1145/3408062
18. Reasoning Using Modular Neural Networks [Electronic resource]. – Access mode: https://towardsdatascience.com/reasoning-using-modular-neural-networks-f003cb6109a2
19. What is a Modular Neural Network? Weak vs. Strong Learners [Electronic resource]. – Access mode: https://caseguard.com/articles/what-is-a-modular-neural-network-weak-vs-strong-learners/
20. Modular neural networks using multiple gradients [Electronic resource]. – Access mode: http://essay.utwente.nl/82225/
21. The CIFAR-10 dataset [Electronic resource]. – Access mode: https://www.cs.toronto.edu/~kriz/cifar.html
22. The CIFAR-100 dataset [Electronic resource]. – Access mode: https://www.cs.toronto.edu/~kriz/cifar.html
23. CIFAR-100 [Electronic resource]. – Access mode: https://paperswithcode.com/dataset/cifar-100
24. The Street View House Numbers (SVHN) Dataset [Electronic resource]. – Access mode: http://ufldl.stanford.edu/housenumbers/
25. The EMNIST Dataset [Electronic resource]. – Access mode: https://www.nist.gov/itl/products-and-services/emnist-dataset
26. ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) [Electronic resource]. – Access mode: https://www.image-net.org/challenges/LSVRC/2012/
27. Caltech 256 Image Dataset [Electronic resource]. – Access mode: https://www.kaggle.com/datasets/jessicali9530/caltech256

DOI 10.15588/1607-3274-2024-2-8