DOI: https://doi.org/10.15276/aait.04.2021.1
UDC 004.8

Binary classification of small satellites telemetry data based on deep learning approach

Vadim Yu. Skobtsov

ORCID: https://orcid.org/0000-0002-8546-0430; vasko_vasko@mail.ru. Scopus Author ID: 8361519700
United Institute of Informatics Problems of the National Academy of Sciences of Belarus, 6, Surganova Str, Minsk, 220012, Belarus
Belarusian State University of Informatics and Radioelectronics, 9, Gikalo Str, Minsk, 220013, Belarus

ABSTRACT

The paper presents solutions to the actual problem of intelligent analysis of telemetry data from small satellites in order to detect its technical states. Neural network models based on modern deep learning architectures have been developed and investigated to solve the problem of binary classification of telemetry data. It makes possible to determine the normal and abnormal state of the small satellites or some of its subsystems. For the computer analysis, the data of the functioning of the small satellites navigation subsystem were used: a time series with a dimension of 121690 × 9. A comparative analysis was carried out of fully connected, one-dimensional convolution and recurrent (GRU, LSTM) neural networks. We analyzed hybrid neural network models of various depths, which are sequential combinations of all three types of layers, including using the technology of adding residual connections of the ResNet family. Achieved results were compared with results of widespread neural network models AlexNet, LeNet, Inception, Xception, MobileNet, ResNet, and Yolo, modified for time series classification. The best result, in terms of classification accuracy at the stages of training, validation and testing, and the execution time of one training and validation epoch, were obtained by the developed hybrid neural network models of three types of layers: one-dimensional convolution, recurrent GRU and fully connected classification layers, using the technology of adding residual connections. In this case, the input data were normalized. The obtained classification accuracy at the training, validation and testing stages was 0.9821, 0.9665, 0.9690, respectively. The execution time of one learning and validation epoch was twelve seconds. At the same time, the modified Inception model showed the best alternative result in terms of accuracy: 0.9818, 0.9694, 0.9675. The execution time of one training and validation epoch was twenty seven seconds. That is, there was no increase in the classification accuracy when adapting the well-known neural network models used for image analysis. But the training and validation time in the case of the best Inception model increased by more than two times. Thus, proposed and analyzed hybrid neural network model showed the highest accuracy and minimum training and validation time in solving the considered problem according to compared with a number of developed and widely known and used deep neural network models.

Keywords: Neural networks; telemetry data; data analysis; fully-connected networks/layers; 1D-convolutional networks/layers; recurrent networks/layers

***For citation*: Skobtsov V. Yu. Binary classification of small satellites telemetry data based on deep learning approach. *Applied Aspects of Information Technology*. 2021; Vol. 4 No. 4: 299–310. DOI: https://doi.org/10.15276/aait.04.2021.1**

INTRODUCTION

One of the most important tasks at all stages of the life cycle of small spacecraft (SS) is the analysis of their telemetry data (TD) about the functioning of the SS in terms of determining their technical state to ensure safe operation and correct control. The relevance is primarily because one of the main reasons for the loss of SS are failures and incorrect operation of the SS.

A large amount of information, arriving from SS and accumulating in specialized databanks, can be effectively used to improve the process of determining the technical state of the small spacecraft and its subsystems.

The functioning data of the small spacecraft, including telemetric data, are heterogeneous irregular multidimensional data.

Therefore relevant is the research, development and application of models that allow you to analyze this kind of data. It gives the ability to extract useful information from them and then build classification and predictive models using them in order to determine the technical state of the SS for making correct control and operational decisions in the process of SS operation.

Methods of machine learning, artificial intelligence and bioinspired models are currently one of the most promising and widely used approaches in data analysis of high-tech systems. Vivid and well-known examples of their application are the developments of such companies as Facebook, Google, Amazon, Yandex, and research centers of Massachusetts Institute of Technology, universities of Cambridge, Stanford, Berkeley, Princeton, Southern California, Montreal, Moscow Institute of Physics and Technology, Higher School of Economics, Bauman Moscow State Technical University.

Today, to solve the listed tasks, to ensure the required degree of autonomy, quality and efficiency of control of such complex objects as small spacecraft, it is necessary to perform complex automation and intellectualization of the estimation processes and multi-model analysis of SS telemetric information data. However, in most cases, in practice, automation is performed, at best, only partially, and much is done often manually, based on heuristic rules [1]. At the same time, in accordance with GOST 1410-002-2010 and the Strategy for digital transformation of the rocket and space industry until 2025 and the prospect until 2030 of the state corporation Roscosmos, an important task is to create a so-called information system on the technical state and the reliability of space complexes (SC) and their constituent products [2, 3].

Thus, the task of intelligent analysis of small spacecraft telemetry data in order to determine the technical state of small spacecraft is relevant and in demand. At the same time, the development and application of methods for analyzing TD SS based on models of artificial intelligence, machine learning and bioinspired systems allows solving the problem at a new scientific and engineering theoretical and applied levels and increasing the efficiency of the control and operational processes for SS ground control systems.

1. STATEMENT OF THE TD SS BINARY CLASSIFICATION PROBLEM

The initial telemetry data is a time series, which can be represented as a matrix $\boldsymbol{X} = (x_{ij})$, where *i–th* row $\boldsymbol{X}_i$ is the analyzed vector of telemetry indicators at the *i-th* moment of time, the *j* index corresponds to the *j-th* indicator of the telemetry at the *i-th* vector $\boldsymbol{X}_i$.

Definition 1. One-dimensional time series $\boldsymbol{X} = (x_1, x_2, \dots, x_T)$ - an ordered set of real values. The length of $\boldsymbol{X}$ is equal to the number of real values *T*.

Definition 2. An *M*-dimensional time series $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_M)$ consists of *M* different one-dimensional time series $\boldsymbol{X}_j \in R^T$.

Obviously that considering TD time series is *M*-dimensional time series $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_M)$, each element of which $\boldsymbol{X}_j$ is a column of the TD matrix $\boldsymbol{X}$ and at the same time a univariate time series describing the behavior of the *j-th* telemetry indicator on an interval of discrete times [1, *T*].

For each vector of telemetry indicators at the *i-th* moment of time $\boldsymbol{X}_i$, a label of the class $y_i \in \boldsymbol{Y}$ is assigned, which characterizes the SS functioning state analyzed basing on the SS telemetry data or its subsystems.

We consider the case of a binary classification, since the final goal is to determine whether the analyzed vector $\boldsymbol{X}_i$ of the *M*-dimensional time series $\boldsymbol{X}$ belongs to the failure-free or failure state. In this case, the number of classes $K = 2$ and, therefore, $\boldsymbol{Y} \in \{0,1\}$, where 0 denotes a failure-free state and 1 - a failure state of the SS system under analysis.

Thus, the task is to find out the classification model of the following mapping:

$$y: \boldsymbol{X} \to \boldsymbol{Y}. \quad (1)$$

To encode class labels, we use One Hot encoding. In this case the vector $\boldsymbol{X}_i$ of the M-dimensional time series $\boldsymbol{X}$ is labelled by the vector $\boldsymbol{Y}_i = (y_{i0}, y_{i1})$ of dimension $K = 2$. The vector $\boldsymbol{Y}_i$ contains only one value 1, which corresponds to the class label 0: (1,0) or 1: (0,1) (Fig.1).

	0	1
11488	1	0
11489	1	0
11490	0	1
11491	0	1
11492	0	1
11493	1	0
11494	1	0
11495	1	0

Fig.1. **Example of One Hot encoding of class labels**
Source: **compiled by the author**

2. MACHINE LEARNING MODELS IN THE CLASSIFICATION PROBLEM OF TIME SERIES

Over the past two decades, time series classification has been considered as one of the most difficult problems in the area of data mining [4, 5]. With the increasing availability of temporal data [6], hundreds of algorithms have been proposed since 2015 [7]. In fact, any classification problem using data that is recorded taking into account some notion of ordering can be viewed as a time series classification problem [8]. Time series are found in many real applications: data processing of electronic medical records, recognition of human activity, classification of acoustic scenes, cybersecurity, SS functioning states analysis according to TD [9, 10], [11, 12], [13, 14].

Recent publications have been focused on the development of ensemble methods [15, 16], [17, 18]. These approaches use either an ensemble of decision trees (random forest) or an ensemble of different

types of discriminant classifiers (support vector machines (SVMs), k-nearest neighbors (kNN) classifiers with several distance functions) on one or more feature spaces. Most of these approaches have a data transformation step that transforms the original time series.

This approach stimulated the development of an ensemble of 35 COTE classifiers (Collective Of Transformation-based Ensembles) [15], which not only combines different classifiers for the same transformation, but instead combines different classifiers for different representations of different time series. In [19, 20], the advantages of COTE were improved using a hierarchical voting system, having received the HIVE-COTE method. HIVE-COTE. HIVE-COTE is currently considered as the leading time series classification algorithm among classical machine learning models when evaluating 85 datasets from the UCR/UEA archive [7]. To achieve high accuracy, HIVE-COTE becomes extremely computationally demanding and impractical for solving real problems of intelligent analysis of big data [7].

This approach requires the training of thirty-seven classifiers, as well as cross-validation of each hyperparameter of these algorithms, which makes it impossible to train this approach in some situations. To emphasize this impossibility, note that one of these thirty-seven classifiers is the Shapelet transformation [16], the time complexity of which is $O(n^2 l^4)$, where *n* is the number of one-dimensional time series in the dataset, and *l* – the length of the time series.

Having analyzed the current state-of-the-art of classical non deep classifier models, we have established the impracticality of advanced approaches in a number of cases of solving the real big data analysis problems for classifying the time series. Therefore, let us focus further on models of deep learning or neural network models, which have been widely used in recent years to solve various problems of big data mining [21]. This motivated their use for solving the problems of time series analysis [8, 11], [22].

3. DEEP LEARNING NEURAL NETWORK MODELS

Artificial neural networks are a convenient and natural basis for representing information models.

Definition 3. An artificial neural network (neural network model, ANN) is a system consisting of a set of elementary processors connected by the type of nodes of a directed graph, called artificial or formal neurons, and capable of generating output information in response to the input action.

Each neuron is characterized by its current state, by analogy with the nerve cells in the brain, which can be excited or inhibited. It has a group of synapses – unidirectional input connections connected to the outputs of other neurons, and also has an axon – an output connection of a given neuron, from which a signal (excitation or inhibition) enters the synapses of the following neurons.

An artificial neuron imitates the properties of a biological neuron. Here, a set of input signals, indicated as $x_i, i = \overline{1,n}$, are fed to an artificial neuron. These input signals, collectively denoted by the vector $X = (x_1, x_2, \dots, x_n)$, correspond to signals arriving at the synapses of a biological neuron. Each synapse is characterized by the magnitude of the synaptic connection or its weight w_i. Every input signal is multiplied by the corresponding weight w_i, and supplied to the adder block. Each weighting factor corresponds to the "strength" of one biological synaptic connection and is analogous to the synapse of biological neurons. If the value of the coefficient is negative, then it is customary to consider the *i-th* connection as inhibitory, if positive – as exciting. The set of weights in the aggregate are denoted by the vector $\vec{w}$. The adder block corresponds to the body of a biological neuron. It adds the weighted inputs algebraically, creating the value S. The resulting value S is fed to the activation output function $a(S)$ of the neuron, simulating the process of activation or inhibition of input impulses or the nonlinear transfer characteristic of a biological neuron.

Thus, the mathematical model of an artificial neuron can be represented by the expression

$$y = a(S) = a(\sum_{i=1}^{n} w_i x_i + b), \quad (2)$$

where: *y* is the output signal of the neuron; *b* – initial bias of the neuron.

In an enlarged form, ANN performs a functional mapping between input and output, and can serve as an information model of the mapping (1). According to [23, 24], the function determined by the neural network can be arbitrary with easily met requirements for the structural complexity of the network and the presence of nonlinearity in the transient (activation) functions of neurons.

Thus, a neural network model consisting of a set of interconnected artificial neurons is a bioinspired model of neural biological systems. At the same time, modern neural network models have found wide application in the theory and practice of data mining and machine learning, including in classification problems of time series of various nature [8, 9], [10, 11], [12, 13], [14], [21, 22].

In the process of development and analysis, we shall consider neural network models from simple to more complex, starting with the basic neural network models now:

– fully connected neural networks/layers (FCNN);

– one-dimensional convolutional neural networks/layers (1D CNN);

– recurrent neural networks/layers (RNN) like *Long Short-Term Memory* (LSTM) and *Gated Recurrent Units* (GRU), and continuing them with combinations of all basic layers, including, based on the method of using the residual connections of the family ResNet.

A comparative analysis will also be carried out with the well-known models AlexNet, LeNet, Inception, Xception, MobileNet, ResNet, Yolo.

Fully connected neural networks/layers can be considered using the example of the following 2-layer network, which is shown in Fig. 2 and is given by the formulas:

$\boldsymbol{X} = (x_1, x_2, x_3)$ – input vector of input layer;

definition of hidden layer:

$\boldsymbol{Z^1} = \boldsymbol{W^1 X} + \boldsymbol{b^1}$; $\boldsymbol{A^1} = a_1(\boldsymbol{Z^1})$;

definition of output layer:

$\boldsymbol{Z^2} = \boldsymbol{W^2 A^1} + \boldsymbol{b^2}$;

$\hat{\boldsymbol{y}} = \boldsymbol{A^2} = a_2(\boldsymbol{Z^2})$,

where: $\boldsymbol{W^i}$ – weighting coefficient values;

$\boldsymbol{b^i}$ – bias values;

$a_i(\boldsymbol{Z^i})$ – activation functions of layers.

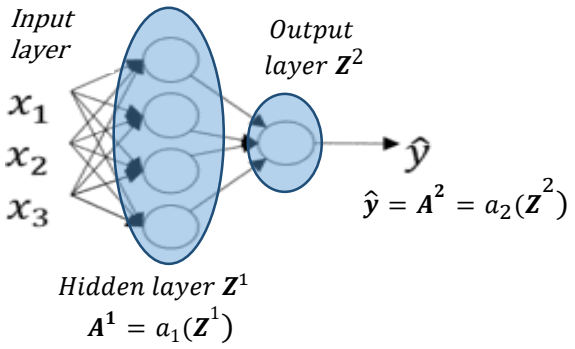

Fig.2. **An example of a two-layer fully-connected neural network**

Source: **compiled by the author**

We shall use the following widely used activation functions in our models [25, 26]:

– function relu – rectified linear unit

$relu\,(z) = max\,(0, z)$;

– generalization of the logistic function for One Hot encoding of class labels

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=0}^{1} e^{z_j}},$$

where: $z_i = W_i Y$ – the value of the *i-th* component of the output layer; Y – the output vector of the previous layer or the input vector in the case of a single-layer neural network or regression model; W_i – vector of weighting coefficients of connections from vector Y to output z_i.

As the loss function, we shall use the binary crossentropy function, since we are solving the binary classification problem [25, 26].

Unlike 2D convolutional neural networks/layers used in image analysis, we will explore one-dimensional convolutional networks/layers [5, 26]. Similar to 2D convolutions that extract 2D templates from image tensors and apply identical transformations to every such template, one-dimensional convolutions can be used to extract one-dimensional templates (subsequences) from time series. This type of one-dimensional convolutional layers are capable of recognizing local patterns in a sequence.

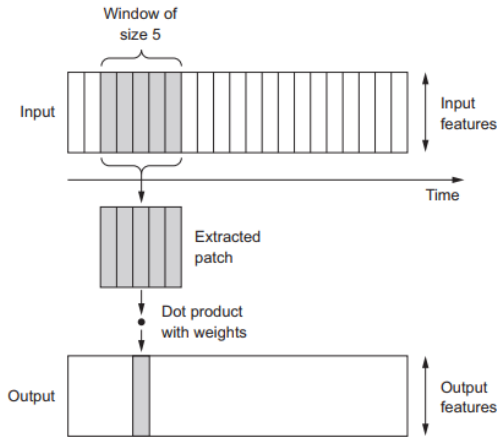

Fig.3. **How a 1D convolutional neural network/layer works**

Source: **compiled by [26]**

In contrast to fully connected networks, the same convolution (the same filter values *w* and *b*) will be used to find the result for all time stamps $t \in [1, T]$. This is a very powerful property of 1D CNN, which allows them to study time-invariant filters. When considering a time series as input to a convolutional layer, the filter no longer has one dimension (time), but also has dimensions that are equal to the number of dimensions in the input time series. Since the same transformations are applied to each template, one or another template found at some position in the sequence can later be recognized at a different position, which makes the transformations performed by 1D CNN networks/layers invariant (in time). For example, a 1D CNN network that processes a sequence of values and uses a convolution window with a size of 5 is able to memorize subsequences of a series of up to 5 elements and recognize them in any context in the input sequence of a time series (Fig. 3).

Instead of manually adjusting the ω filter values, these values should be learned automatically as they are highly dependent on the target dataset. For example, one dataset will have an optimal filter of (1, 2, 2), while another dataset will have an optimal filter of (2, 0, -1). By optimal, we mean a filter, the application of which will allow the classifier to easily distinguish between the classes of the data set [5, 26].

The information extracted by the convolution is fed, as in the case of a fully connected ANN, to the input of the activation function $a(\boldsymbol{Z})$. A block of 1D CNN layers must be followed by a discriminant classifier, which is usually a block of fully connected layers. It can be preceded by an aggregation operation (Pooling), which can also be present as an intermediate layer between 1D CNN blocks of layers. Pooling average (AveragePooling) or maximum (MaxPooling) takes an input time series and reduces its length T by aggregating in a sliding window of the time series. For example, if the length of the sliding window is 3, the resulting merged time series will have a length equal to *T*/3 (this is true only if the step is equal to the length of the sliding window). The essence of aggregation is element-wise multiplication in a sliding window of a subsequence by a mask, calculating the average or maximum value and replacing the subsequence with it.

A distinctive feature of the neural networks/layers that we have looked at is the lack of memory. Each input is processed independently without saving the state between them in this case. A recurrent neural network processes a sequence, iterating over its elements and preserving the state obtained when processing previous elements. In fact, RNN is a kind of neural network with an internal state [25, 26] (Fig. 4).

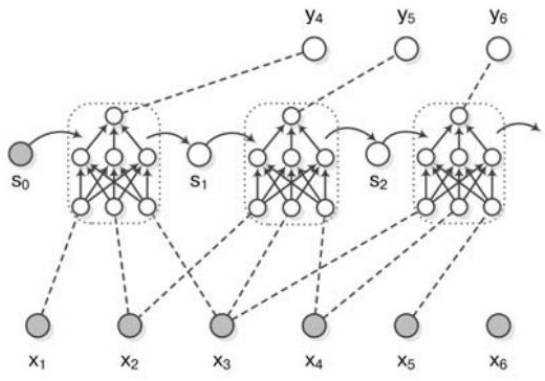

Fig.4. **General diagram of a recurrent neural network/layer**
Source: **compiled by [25]**

One of the well-known RNN models is the LSTM RNN (Long Short-Term Memory). The LSTM cell is shown in Fig. 5 and consists of three main gate nodes: an input gate, a logic gate, and an output gate, which form a recurrent cell with a hidden state.

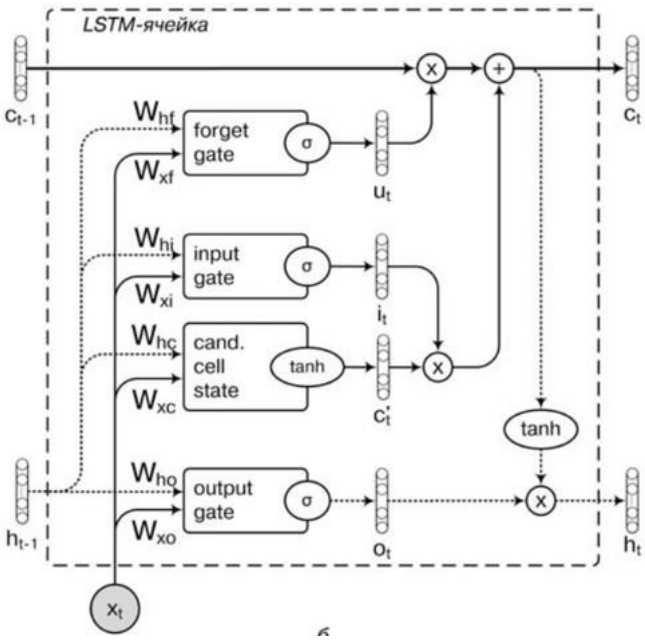

Fig.5. **LSTM cell structure**
Source: **compiled by [25]**

At the same time, RNN resembles micro-electronic sequential circuits with memory, decomposed into its combinational iterative equivalent.

If we denote by x_t input vector at time *t*, h_t – hidden state vector at time *t*, weight matrices:
in $candidate\ cell\ state$:
W_{xc} – from the input vector x_t, W_{hc} – from the hidden state vector at the moment of time *t*-1;
in $input\ gate$:
W_{xi} – from the input vector x_t, W_{hi} – from the hidden state vector at the moment of time *t*-1;
in $forget\ gate$:
W_{xf} – from the input vector x_t, W_{hf} – from the hidden state vector at the moment of time *t*-1;
in $output\ gate$:
W_{xo} – from the input vector x_t, W_{ho} – from the hidden state vector at the moment of time *t*-1,
$b_{c\prime}, b_i, b_f, b_o$ – vectors of biases in cells $candidate\ cell\ state$, $input\ gate$, $forget\ gate$, $output\ gate$ accordingly,
we get the following formal definition of how the LSTM works on the next input x_t, having the hidden state from the previous step h_{t-1} and the actual state of the cell c_{t-1}, we sequentially calculate [25]:

$$c'_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_{c'}) \quad candidate\ cell\ state,$$

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad input\ gate,$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad forget\ gate,$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad output\ gate,$$

$$c_t = f_t \circ c_{t-1} + i_t \circ c'_t \quad cell\ state,$$

$h_t = o_t \circ \tanh(c_t)$ *block output*.

In work [27] in 2014, a modification of LSTM recurrent networks – Gated Recurrent Unit (GRU) was proposed, which reduced the complexity of LSTM model and training time. In this architecture, the hidden state h_t aligned with memory value c_t.

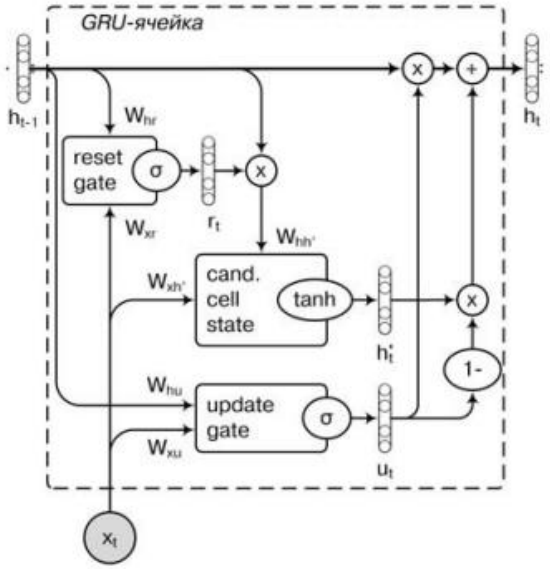

Fig.6. **GRU cell structure**
Source: **compiled by [25]**

This is how a single GRU cell works [25]:

$u_t = \sigma(W_{xu}x_t + W_{hu}h_{t-1} + b_u)$,
$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$,
$h_t' = \tanh(W_{xh'}x_t + W_{hh'}(r_t \circ h_{t-1}))$,
$h_t = (1 - u_t) \circ h_t' + u_t \circ h_{t-1}$.

Here u_t – update gate; r_t – reset gate, he is also responsible for what part of memory needs to be transferred further from the last step, but he does this even before the nonlinear function is applied. Memory cell and block output h_t in this case, unlike LSTMs, are not separated in any way, and the next output h_t obtained as a combination (set by the gate u_t) previous output h_{t-1} and the current output candidate h'_t, which, in turn, also depends on h_{t-1}, but through the reset gate r_t (Fig.6).

4. DEVELOPMENT AND ANALYSIS OF NEURAL NETWORK MODELS

Based on the neural network architectures described above, the following neural models have been proposed and investigated. The developed models were implemented in Python using the Keras and Tensorflow deep learning libraries (other libraries necessary for processing and visualizing data were also used: numpy, matplotlib, pandas and scikit-learn). Accordingly, the pseudocode of the given models is based and close to the Keras/Tensorflow notation for better reproducibility.

Fully connected neural network model:

$\boldsymbol{Z}$ = Dense(16, activation='relu') ($\boldsymbol{X}_i$)
$\boldsymbol{Z}$ = Dense(16, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Dense(2, activation='softmax') ($\boldsymbol{Z}$),
where Dense – fully connected layer notation [25,26].

Convolutional 1D CNN model:

$\boldsymbol{Z}$ = Conv1D(filters=256, kernel_size=4, activation='relu') ($\boldsymbol{X}_i$)
$\boldsymbol{Z}$ = MaxPooling1D(2) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Conv1D(filters=128, kernel_size=2, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = GlobalMaxPooling1D ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Dense(2, activation='softmax') ($\boldsymbol{Z}$)

Recurrent LSTM model:

$\boldsymbol{Z}$ = LSTM(units=64) ($\boldsymbol{X}_i$)
$\boldsymbol{Z}$ = LSTM(units=32) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = LSTM(units=16) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Dense(2, activation='softmax') ($\boldsymbol{Z}$)

Recurrent GRU model:

$\boldsymbol{Z}$ = GRU (units=64) ($\boldsymbol{X}_i$)
$\boldsymbol{Z}$ = GRU (units=32) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = GRU (units=16) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Dense(2, activation='softmax') ($\boldsymbol{Z}$)

Computer analysis was carried out on real telemetry data of one of the navigation subsystems of the small spacecraft. Each vector of the TD matrix $\boldsymbol{X}_i$ has a dimension of 9 and is labelled 0 in the case of a free-failre state and 1 in the case of failre state of the subsystem. The total dimension of the 9-dimensional time series $\boldsymbol{X}$ is 121,690 vectors, 77881 vectors make up the training dataset, 19471 vectors make up the validation dataset, 24338 vector make up the test dataset.

For the above group of neural network models, training and validation were carried out with the following values of hyperparameters: the "adam" training method (as one of the most effective at the moment), the loss function "binary_crossentropy", the number of training epochs – 500, mini-batch size – 128. The early stopping mechanism [26] was not used and the learning and validation process took place at all 500 epochs. The results of computer experiments including the value of accuracy and loss function on the training, validation and training sets, as well as the time for one epoch, are shown in Table 1.

Table 1. **Experimental data of non-hybrid neural network models**

NN type	Training		Validation		Testing		Time of one training & validation epoch, sec
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	
Fully-connected NN	0.8816	0.2831	0.8810	0.2835	0.8809	0.2851	3
Convolutional neural network 1D CNN	0.9065	0.2206	0.9037	0.2336	0.8999	0.2654	3
Recurrent LSTM NN	0. 9617	0. 0912	0. 9487	0. 1408	0.9485	0.1425	31
Recurrent GRU NN	0.9485	0.1215	0.9358	0.1633	0.9336	0.1723	26

Source: **compiled by the author**

As the data in Table 1 show, a fully connected model has the least accuracy at the training, validation and testing stages, while its training and validation time is the smallest. A more accurate model (> 0.9) is a 1D CNN model, and the training and validation time is equal to the time of the fully connected model. Recurrent models are obviously the leaders in accuracy, and the GRU model at the stage of training, validation and testing is not much inferior to LSTM. At the same time, in terms of training and validation time, the GRU model rather outperforms the LSTM. Therefore, we draw a conclusion about the leadership of the LSTM model in terms of accuracy in this series of experiments. If the accuracy of the model is enough to have more than 0.9 and the factor of training time and model lightness is important, then the 1D CNN convolutional model is more attractive. An increase in the number of layers and neurons in the layers of models did not lead to an increase in the quality of the models. The opposite effect of reaching a plateau and a decrease in accuracy in the learning process was often observed.

The further goal of the research was, on the one hand, to increase the accuracy of the model, on the other hand, to reduce its training and validation time, that is, to obtain a lighter-weight model compared to recurrent ones. For this purposes, we consider hybrid models consisting the three blocks of layers: convolutional Conv1D, recurrent GRU or LSTM, and as a result, as a discriminant classifier, a fully connected block:

$\boldsymbol{Z}$ = Conv1D(filters=512, kernel_size=4, activation='relu') ($\boldsymbol{X}_i$)
$\boldsymbol{Z}$ = Conv1D(filters=512, kernel_size=4, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Conv1D(filters=512, kernel_size=4, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Pooling1D(2) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Conv1D(filters=256, kernel_size=2, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Conv1D(filters=256, kernel_size=2, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Conv1D(filters=256, kernel_size=2, activation='relu') ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = RNN(units=64) ($\boldsymbol{Z}$)
$\boldsymbol{Z}$ = Dense(2, activation='softmax') ($\boldsymbol{Z}$)

Based on this architecture, several neural network models were obtained by using the AveragePooling and MaxPooling methods in the aggregation layer; in the recurrent layers the cells of the GRU and LSTM types were used. The input time series with initial and normalized values in the range from 0 to 1 was also considered, using the MinMaxScaler function. The training was also carried out at 500 epochs, but the mechanism of early stopping was used in case of reaching a plateau of the validation accuracy within 10 iterations. Experiments have shown that in this case the duration of training and validation was no more than 160 epochs. In the recurrent layer, l_2 regularization was used to aliminate overfitting [25,26].

The results of computer experiments representing the value of accuracy and loss functions on the training, validation and test sets are shown in Table 2. Also time of one training and validation epoch is given.

Based on the data in Table 2, in terms of accuracy and time of one training and validation epoch, in this group of models, the model with AveragePooling and GRU cell is leading with a slight advantage.

The next group of models was built on the basis of the architecture of the previous model and the method of adding residual connections. Development of this method began with appearing the ResNet family of networks, developed by Kaiming He and colleagues at Microsoft [26, 28] (Fig. 7).

Table 2. **Experimental data of hybrid sequential models**

NN type	Training		Validation		Testing		Time of one training & validation epoch, sec
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	
AveragePooling, GRU	0.9850	0.0333	0.9680	0.1364	0.9661	0.1327	14
MaxPooling, GRU	0.9810	0.0488	0.9588	0.1557	0.9604	0.1413	14
AveragePooling, LSTM	0.9816	0.0464	0.9649	0.1195	0.9655	0.1188	17
MaxPooling, LSTM	0.9808	0.0454	0.9629	0.1255	0.9642	0.1202	17
AveragePooling, GRU, normalized data	0.9796	0.0452	0.9675	0.1138	0.9659	0.1039	15
MaxPooling, GRU, normalized data	0.9789	0.0439	0.9686	0.1077	0.9674	0.1151	15
AveragePooling, LSTM, normalized data	0.9791	0.0447	0.9662	0.1070	0.9671	0.1022	16
MaxPooling, LSTM, normalized data	0.9765	0.0572	0.9640	0.1028	0.9661	0.0942	17

Source: **compiled by the author**

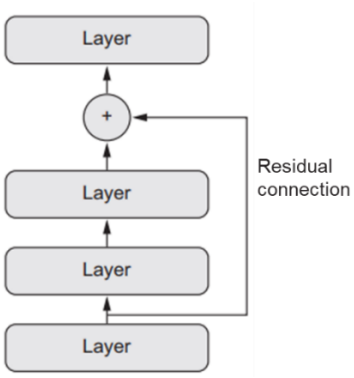

Fig.7. **Residual connection: reinjection of prior information by adding to the feature map of later layers**

Source: **compiled by [26]**

Also, the number of convolutional layers has been increased and the number of filters in them has been reduced.

$\boldsymbol{Z}_1$ = Conv1D(filters=64, kernel_size=4, activation='relu') ($\boldsymbol{X}_i$)

$\boldsymbol{Z}_1$ = Conv1D(filters=64, kernel_size=4, activation='relu') ($\boldsymbol{Z}_1$) * 9 слоев

$\boldsymbol{Z}_2$ = add([$\boldsymbol{Z}_1$, $\boldsymbol{X}_i$]) – residual connection $\boldsymbol{X}_i$

$\boldsymbol{Z}_2$ = Pooling1D(2)($\boldsymbol{Z}_2$)

$\boldsymbol{Z}_3$ = Conv1D(filters=64, kernel_size=2, activation='relu') ($\boldsymbol{Z}_2$)

$\boldsymbol{Z}_3$ = Conv1D(filters=64, kernel_size=2, activation='relu') ($\boldsymbol{Z}_2$) * 9 слоев

$\boldsymbol{Z}_4$ = Pooling1D(2)($\boldsymbol{X}_i$)

Output = add([$\boldsymbol{Z}_2$, $\boldsymbol{Z}_3$, $\boldsymbol{Z}_4$]) – residual connections $\boldsymbol{Z}_2$ and $\boldsymbol{Z}_3$

Output = GRU(units=32) (***Output***)

Output = Dense(32, activation='relu') (***Output***)

Output = Dense(2, activation='softmax') (***Output***)

The results of computer experiments representing the value of accuracy and loss function on the training, validation and test sets, as well as the time of one training and validation epoch, are given in Table 3.

Table 3. **Experimental data of hybrid models using residual connections**

NN type	Training		Validation		Testing		Time of one training & validation epoch, sec
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	
AveragePooling, GRU	0.9787	0.0844	0.9610	0.1389	0.9584	0.1483	13
MaxPooling, GRU	0.9816	0.0667	0.9619	0.1331	0.9624	0.1334	12
AveragePooling, LSTM	0.9791	0.0629	0.9574	0.1360	0.9570	0.1318	13
MaxPooling, LSTM	0.9784	0.0926	0.9587	0.1451	0.9580	0.1524	13
AveragePooling, GRU, normalized data	0.9821	0.0625	0.9665	0.1091	0.9690	0.1075	12
MaxPooling, GRU, normalized data	0.9799	0.0736	0.9646	0.1213	0.9663	0.1154	12
AveragePooling, LSTM, normalized data	0.9726	0.0970	0.9573	0.1445	0.9581	0.1408	12
MaxPooling, LSTM, normalized data	0.9630	0.1166	0.9531	0.1428	0.9539	0.1403	12

Source: **compiled by the author**

According to Table 3, based on the ratio of the accuracy value and the loss function, at the training, validation and testing stages, as well as the time of one training and validation epoch, the leader is a model with the parameters: AveragePooling, GRU and normalized data.

Further, the computer analysis results of the application of widely known and used neural network models AlexNet, LeNet, Inception, Xception, MobileNet, ResNet, Yolo was carried out for the problem under consideration. As a basis, we used the program code from the repository of the series of online workshops "Machine Learning Tokyo - Democratizing Machine Learning" [29], which was modified for the task of classifying the TD time series of the analyzed SS subsystem. The results of computer experiments, representing the value of accuracy and loss function on the training, validation and test sets, as well as the time of one training and validation epoch, are given in Table 4.

Based on the data in Table 4, we can conclude that the best results in terms of classification accuracy values and execution time of one training and validation epoch were shown by a modified deep neural network model of the Inception family, developed by Christian Szegedy and colleagues at Google [26, 30]. It used the AveragePooling aggregation layer, the LSTM recurrent layer, and data normalization. There was no significant increase in the values of accuracy and decrease in the values of losses at the stages of validation and testing for the last group of models. At the same time, the time of one epoch of training and validation has increased more than 2 times. In addition, the Inception model has become significantly more complicated in comparison with the best-developed model from Table 3. Therefore, to solve the problem under consideration, the use of this group of models, unfortunately not advisable.

CONCLUSION

Neural network models based on modern deep learning architectures have been developed and investigated to solve the problem of binary classification of telemetry data, which make it possible to determine the normal and abnormal state of functioning of the SS navigation subsystem. A computer analysis was carried out on the real TD, which made it possible to assess the quality of the developed models at the stages of training, validation and testing. This analysis showed the advantage of hybrid neural network models, which are a sequential connection of three blocks of layers: convolutional 1D CNN, recurrent GRU and the final fully connected classifier block, using the AveragePooling aggregation layer, the method of the residual connections and normalizing the initial data. A similar model also performed well without residual connections and input data normalization. In general, when solving the problem under consideration, an accuracy of more than 0.96 at the stages of validation and testing was achieved.

Further research of modern neural network models for the problem of binary and non-binary classification of time series, as well as methods for automating the search for optimal hyperparameters values of models and their architectures, is promising.

Table 4. **Experimental data of modified neural network models**

NN type	Training		Validation		Testing		Time of one training & validation epoch, sec
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	
AlexNet	0.9221	0.1772	0.9215	0.1817	0.9228	0.1782	22
LeNet	0.8690	0.3011	0.8669	0.3062	0.8695	0.3072	10
Inception	0.9818	0.0406	0.9694	0.1045	0.9675	0.1154	27
Xception	0.9226	0.1749	0.9252	0.1748	0.9216	0.1755	96
MobileNet	0.9650	0.0776	0.9608	0.0949	0.9597	0.0975	35
ResNet	0.9685	0.0687	0.9656	0.0969	0.9644	0.0924	79
Yolo	0.9359	0.1665	0.9302	0.1815	0.9322	0.1783	37

Source: **compiled by the author**

REFERENCES

1.Ohtilev, M. Y U., Mustafin, N. G., Miller, V. E. & Sokolov, B. V. "The concept of proactive management of complex objects: theoretical and technological foundations" (in Russian). *Journal of Instrument Engineering*. 2014; Vol. 57 No.11: 7–14.

2. GOST RO 1410-002-2010. "Rocket and space technology. Information system about the technical condition and reliability of space complexes and their products" (in Russian).

3."Project of information technology strategy of the State Corporation "Roscosmos"" (in Russian). – Available from: https://www.roscosmos.ru/25892/. – [Accessed: 15 Nov. 2020].

4. Yang, Q. & Wu, X. "Ten challenging problems in data mining research". *International Journal of Information Technology & Decision Making.* 2006; Vol.5 No. 04: 597–604. DOI: https://doi.org/10.1142/S0219622006002258.

5. Ismail Fawaz, H., Forestier, G., Weber, J. & Idoumghar, L. "Pierre-alain muller deep learning for time series classification: a review". *Data Mining and Knowledge Discovery*. 2019; 33: 917–963. DOI: https://doi.org/10.1007/s10618-019-00619-1.

6. Silva, D. F., Giusti, R., Keogh, E. & Batista, G. "Speeding up similarity search under dynamic time warping by pruning unpromising alignments". *Data Min Knowl Discov*. 2018; 32(4); 988–1016. DOI: https://doi.org/10.1007/s10618-018-0557-y.

7. Bagnall, A., Lines, J., Bostrom, A., Large, J. &, Keogh, E. "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances". *Data Min Knowl Discov*. 2017; 31(3): 606–660. DOI: https://doi.org/ 10.1007/s10618-016-0483-9.

8.Cristian Borges Gamboa J. "Deep learning for time-series analysis". 2017. – Available from: arXiv:1701.01887. – [Accessed: 15 Nov. 2020].

9. Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Liu, P. J., Liu, X., Sun, M., Sundberg, P., Yee, H., Zhang, K., Duggan, G. E., Flores, G., Hardt, M., Irvine, J., Le, Q., Litsch, K., Marcus, J., Mossin, A., Tansuwan, J., Wang, D., Wexler, J., Wilson, J., Ludwig, D., Volchenboum, S. L., Chou, K., Pearson, M., Madabushi, S., Shah, N. H., Butte, A. J., Howell, M., Cui, C., Corrado, G. & Dean, J. "Scalable and accurate deep learning with electronic health records". *NPJ Digit Med.* 2018. p.1:18. DOI: https://doi.org/ 10.1038/s41746-018-0029-1.

10. Nweke, H. F., Teh, Y. W., Al-garadi, M. A. & Alo, U. R. "Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: state of the art and research challenges". *Expert Syst Appl.* 2018; 105: 233–261. DOI: https://doi.org/10.1016/j.eswa.2018.03.056.

11. Nwe, T. L, Dat, T. H. & Ma, B. "Convolutional neural network with multi-task learning scheme for acoustic scene classification". *In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference.* 2017. p.1347–1350. DOI: https://doi.org/10.1109/APSIPA.2017.8282241.

12. Susto, G. A., Cenedese, A. & Terzi, M. "Time-series classification methods: review and applications to power systems data". *In: Big Data Application in Power Systems.* 2018. p.179–220. DOI: https://doi.org/10.1016/B978-0-12-811968-6.00009-7.

13. Skobtsov, V., Novoselova, N., Arhipov, V. & Potryasaev, S. "Intelligent telemetry data analysis of small satellites" In: Silhavy R., Senkerik R., Kominkova Oplatkova Z., Prokopova Z., Silhavy P. (eds) "Cybernetics and Mathematics Applications in Intelligent Systems". *CSOC 2017. Advances in Intelligent Systems and Computing – Springer*. 2017; Vol. 574: 351–361. DOI: https://doi.org/10.1007/978-3-319-57264-2_36.

14. Skobtsov, V. Yu. & Arhipau, V. I. "Neural network analysis of telemetry data of on-board equipment of spacecraft". *Space Engineering and Technology.* 2021; No. 3(34): 111–124 (in Russian). DOI: https://doi.org/10.33950/spacetech-2308-7625-2021-3-111-124.

15. Bagnall, A., Lines, J., Hills, J. & Bostrom, A. "Time-series classification with COTE: the collective of transformation-based ensembles". *In: International Conference on Data Engineering*. 2016. p.1548–1549. DOI: https://doi.org/10.1109/ICDE.2016.7498418.

16. Hills, J., Lines, J., Baranauskas, E., Mapp, J. & Bagnall, A. "Classification of time series by shapelet transformation". *Data Min Knowl Discov.* 2014; 28(4): 851–881. DOI: https://doi.org/10.1007/s10618-013-0322-1.

17. Bostrom, A. & Bagnall, A. "Binary shapely transform for multiclass time series classification. *In: Big data analytics and knowledge discovery"*. 2015. p. 257–269. DOI: https://doi.org/10.1007/978-3-662-55608-5_2.

18. Baydogan, M. G. "Multivariate time series classification datasets". 2015. – Available at: http://www.mustafabaydogan.com. – [Accessed: 15 Nov. 2020].

19. Lines, J., Taylor, S. & Bagnall, A. “HIVE-COTE: the hierarchical vote collective of transformation-based ensembles for time series classification”. *In: IEEE International Conference on Data Mining*. 2016. p.1041–1046. DOI: https://doi.org/10.1109/ICDM.2016.0133.

20. Lines, J., Taylor, S. & Bagnall, A. “Time series classification with HIVE-COTE: the hierarchical vote collective of transformation-based ensembles”. *ACM Trans Knowl Discov Data*. 2018; 12(5): 52:1–52:35. DOI: https://doi.org/10.1145/3182382.

21. LeCun, Y., Bengio, Y. & Hinton, G. “Deep learning”. *Publ. Nature.* 2015; 521: 436–444. DOI: http://dx.doi.org/10.1038/nature14539.

22. Wang, Z., Yan, W. & Oates, T. “Time series classification from scratch with deep neural networks: a strong baseline”. *In: International Joint Conference on Neural Networks*. 2017. p.1578–1585. DOI: http://dx.doi.org/10.1109/IJCNN.2017.7966039.

23. Gorban, A. N., Dunin-Barkovsky, V. L., Kirdin, A. N. et al. “Neuroinformatics” (in Russian). *Publ. Nauka*. Novosibirsk: Russian Federation. 1998. 296 p.

24. Tsaregorodtsev, V. G. “Neuroimitator NEUROPRO” (in Russian). Neuroinformatics and its applications: Book of abstracts of VI All-Russian workshop. KSTU. Krasnoyarsk: Russian Federation. October 2-5, 1998. 207 p.

25. Nikolenko, S., Kadurin, A. & Arkhangelskaya, E. “Deep learning” (in Russian). SPb.: Piter. Russian Federation 2018. 480 p.

26. Chollet, F. “Deep Learning with Python”. *Manning Publ*. Shelter: Island, NY. 2018. 384 p.

27. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. 2014. – Available at: arXiv: 1406.1078. – [Accessed: 15 Nov. 2020].

28. Kaiming, He., et al. “Deep Residual Learning for Image Recognition”. *Conference on Computer Vision and Pattern Recognition*. 2015. – Available at: https://arxiv.org/abs/1512.03385. – [Accessed: 15 Nov. 2020].

29. “Machine Learning Tokyo – Democratizing Machine Learning”. – Available from: https://github.com/Machine-Learning-Tokyo/DL-workshop-series/blob/master/Part%20I%20%20Convolution%20Operations/ConvNets.ipynb. – [Accessed: 15 Nov. 2020].

30. Szegedy, C. et al. “Going Deeper with Convolutions”. *Conference on Computer Vision and Pattern Recognition*. 2014. – Available from: https://arxiv.org/abs/1409.4842. – [Accessed: 15 Nov 2020].

Conflicts of Interest: The authors declare no conflict of interest

Received 09.01.2021
Received after revision 12.03.2021
Accepted 15.03.2021

DOI: https://doi.org/10.15276/aait.04.2021.1

УДК 004.8

Бінарна класифікація даних телеметричної інформації малих космічних апаратів на основі глибокого навчання

Вадим Юрійович Скобцов
ORCID: https://orcid.org/0000-0002-8546-0430; vasko_vasko@mail.ru. Scopus Author ID: 8361519700
Об’єднаний інститут проблем інформатики НАН Білорусії, вул. Сурганова, 6. Мінськ, 220012, Білорусь
Білоруський державний університет інформатики та радіоелектроніки. вул. Гікало, 9, Мінськ, 220013, Білорусь

АНОТАЦІЯ

У статті надано розв’язання актуальної задачі інтелектуального аналізу даних телеметричної інформації малих космічних апаратів з метою визначення їх технічних станів. Розроблено нейромережеві моделі на основі сучасних архітектур глибокого навчання для вирішення задачі бінарної класифікації даних телеметричної інформації, що дозволяють визначати штатний та позаштатний стан функціонування малих космічних апаратів або деяких їх підсистем. Для комп'ютерного аналізу використовувалися дані функціонування навігаційної підсистеми малих космічних апаратів, часовий ряд розмірністю 121690×9. Проводився порівняльний аналіз повнозв'язних, одновимірних згорткових та рекурентних (GRU, LSTM) нейронних мереж, нейронних моделей різної глибини, які є послідовними комбінаціями всіх трьох типів шарів, у тому числі з використанням технології додавання залишкових зв'язків сімейства ResNet, широко поширених нейромережних моделей AlexNet, LeNet , Inception, Xception, MobileNet, ResNet, Yolo, що є модифікованими для класифікації часових рядів. Найкращий результат з точки зору точності класифікації на етапах навчання, валідації, тестування, та часу виконання однієї епохи навчання та валідації отримали розроблені послідовні нейромережеві моделі з трьох типів шарів: одновимірних згорткових, рекурентного GRU та повнозв’язкового класифікаційного шарів. Вхідні дані було внормовано. Точність класифікації на етапах навчання, валідації та тестування склали відповідно: 0.9821, 0.9665, 0.9690. Час виконання однієї епохи навчання та валідації склав дванадцять сек. При цьому найкращий альтернативний результат показала модифікована модель Inception: 0.9818, 0.9694, 0.9675. Час виконання однієї епохи навчання та валідації для цієї моделі склав двадцять сім сек. Збільшення точності класифікації під час адаптації відомих нейромережевих моделей, які використовуються для аналізу зображень, отримано не було, але час навчання та валідації у разі кращої моделі Inception збільшився більш ніж у два рази. Були запропоновані та проаналізовані гібридні нейромережеві моделі, у тому числі з використанням методики прокидання залишкових зв'язків сімейства ResNet. Вони показали найбільшу точність та мінімальний час навчання та валідації моделі у вирішенні поставленої задачі порівняно з низкою розроблених та широко відомих, застосовуваних глибоких нейромережевих моделей.

Ключові слова: нейронні мережі; дані телеметрії; аналіз даних; повнозв’язкові мережі/шари; 1D-згорткові мережі/шари; рекурентні мережі/шари

ABOUT THE AUTHOR

Vadim Yu. Skobtsov – PhD (Eng), Associate Professor, Leading Researcher at the United Institute of Informatics Problems of the National Academy of Sciences of Belarus, 6, Surganova Str, Minsk, 220012, Belarus
Doctoral Student of the Department of Information Technology Software of the Belarusian State University of Informatics and Radioelectronics, 9, Gikalo Str. Minsk, 220013, Belarus
ORCID: https://orcid.org/0000-0002-8546-0430; vasko_vasko@mail.ru. Scopus Author ID: 8361519700
***Research field*:** Intelligent data analysis; machine learning; soft computing; logical-probabilistic modeling and their applications in the field of reliability; security and risk assessment of hardware and software systems; image data analysis; information system and security models based on private blockchain; test diagnostics and simulation of digital systems

Вадим Юрійович Скобцов – кандидат технічних наук, доц., провідний науковий співробітник Об’єднаного інституту проблем інформатики Національної академії наук Білорусії, вул. Сурганова, 6. Мінськ, 220012, Білорусь.
Докторант кафедри Програмного забезпечення інформаційних технологій Білоруського державного університету інформатики та радіоелектроніки, вул. Гікало, 9. Мінськ, 220013, Білорусь