Neural network method for identifying potential defects in complex dynamic objects

Victoria Vysotska[1,†], Vasyl Lytvyn[1,†], Serhii Vladov[2,*,†], Ruslan Yakovliev[2,†] and Yevhen Volkanin[2,†]

[1] *Lviv Polytechnic National University, Stepan Bandera Street 12 79013 Lviv, Ukraine*
[2] *Kremenchuk Flight College of Kharkiv National University of Internal Affairs, Peremohy Street 17/6 39605 Kremenchuk, Ukraine*

Abstract

The work is devoted to the development of a neural network method for identifying potential defects in complex dynamic objects, such as, for example, helicopter turboshaft engines. The proposed method is based on the use of the Transformer model, consisting of the encoder, decoder, positional encoding, and attention mechanism, instead of a generalized regression neural network. A modification of the ReLU activation function in the form of Smooth ReLU is proposed to make it smoother and more continuous, which leads to improved convergence and training stability. The analysis of the derivatives of the ReLU and Smooth ReLU functions showed that Smooth ReLU solves the problem of "dead neurons" by providing a non-zero gradient for all input data values, including negative ones, which ensures more stable training of neural networks and prevents neurons from stopping updating due to a zero gradient. As a neural network implementation of the Transformer model, the use of a graph neural network is proposed, the key advantage of which is its ability to model more complex dependencies and relationships between input data elements, which increases the efficiency of training and improves the quality of prediction in sequence processing tasks. As a neuron activation function, the use of a cross-entropy loss function between actual and predicted probability distributions has been proposed, the key advantage of which is its ability to provide efficient training of a classification model by minimizing the discrepancy between predicted and real class probabilities. The results of the computational experiment showed that the proposed method demonstrated almost 100 % accuracy in determining potential defects, such as the possible formation of cracks (burnouts) in the combustion chamber of helicopter turboshaft engines due to the predicted decrease in the gas temperature in front of the compressor turbine.

Keywords

Neural network, Transformer architecture, graph neural network, training, identifying potential defects, helicopter turboshaft engine, activation function, adaptive training rate

CITI'2024: 2nd International Workshop on Computer Information Technologies in Industry 4.0, June 12–14, 2024, Ternopil, Ukraine
* Corresponding author.
† These authors contributed equally.
victoria.a.vysotska@lpnu.ua (V. Vysotska); vasyl.v.lytvyn@lpnu.ua (V. Lytvyn); serhii.vladov@univd.edu.ua (S. Vladov); director.klk.hnuvs@gmail.com (R. Yakovliev); volkanin@ukr.net (Y. Volkanin)
0000-0001-6417-3689 (V. Vysotska); 0000-0002-9676-0180 (V. Lytvyn); 0000-0001-8009-5254 (S. Vladov); 0000-0002-3788-2583 (R. Yakovliev); 0000-0003-3507-1987 (Y. Volkanin)

1. Introduction

Currently, the use of neural networks is becoming increasingly common in various fields, including the prediction and identification of potential defects in various systems and processes [1–3]. Neural network methods are a powerful tool for analyzing data and identifying hidden patterns [4, 5], which makes them attractive for use in tasks of prediction and defect detection.

Neural networks are mathematical models that attempt to imitate the functioning of the human brain. They consist of many interconnected neurons organized into layers that process input data and generate a corresponding output [6, 7]. In predicting tasks, neural networks can be used to analyze time series, process images, texts, and other types of data [8–10].

Identifying potential defects is an important step in the process of quality control and ensuring the reliability of systems and equipment. Neural network methods can be effectively applied to detect anomalies and predict possible failures based on historical data about the operation of a system or process [1, 11, 12].

The work aims to develop a neural network method for identifying potential defects in various systems and processes using data analysis and identifying hidden patterns to increase the efficiency of quality control and ensure the reliability of equipment and systems.

2. Related works

The neural network method for identifying potential defects based on prediction results is a powerful tool for increasing the efficiency of quality control and ensuring the reliability of systems and processes [13, 14]. The use of neural network methods allows you to automate the process of detecting anomalies and predicting failures, which helps improve production processes and reduce the cost of equipment maintenance and repair.

The significance of this research area allows us to expand the capabilities of diagnostic models and increase the reliability of diagnosis. In [15, 16], the advantages of using artificial intelligence methods, including neural networks, fuzzy logic, and expert systems, compared to traditional diagnostic methods to solve issues related to predicting the reliability of complex dynamic systems were identified. This is because artificial intelligence systems are highly adaptable and capable of solving complex tasks of classification and pattern recognition, which is an important aspect of the process of diagnosis and prognosis.

For example, in [17], the task of the YOLOv5 network model structure optimization using the Convolutional Block Attention Module (CBAM) was solved to improve the accuracy of detection and identification of small defects in pipeline circumferential welds based on the analysis of internal magnetic flux (MFL) signals, which helps to improve technical support and safety in assessing the condition and repair of pipelines. The disadvantage of [15] is that the YOLOv5 model, although effective in detecting anomalous objects, does not exhibit attentional preference during the feature extraction process, which is insufficient for the effective detection of small defects.

The essence of [18] is to develop and experimentally validate a wavelet-gradient-integrated neural network (WGI-SNN) structure combining wavelet packet transform (WPT) with peak neural network (SNN) to more effectively measure complex equipment defects using frequency-domain technologies. -temporal signal processing. A disadvantage of the work is the limited experimental testing of the proposed method, which may reduce the generalizability of the results to various types of complex defects.

In [19] presents a neural network method for detecting binary and random impulse noise in contaminated images, based on pixel classification and random point patterning, which applies to colour images and demonstrates superiority over many existing methods in extensive experimental results. The key disadvantage of the proposed method is its dependence on the classification accuracy of random point patterns and the possibility of isolated points being misidentified as noise, which can lead to the loss of important information in the images.

In [20] it proposed a method that uses multi-threshold binarization of multispectral images to extract features and improve classification accuracy in real-time, reducing training and inference time by 5 times compared to ResNet and Ensemble CNN models. The key disadvantage is its limitation in working with small data sets and possible reduction in classification accuracy when using large data sets.

Based on the discussion of related works, it becomes urgent to develop methods that, based on the analysis of process behaviour and the results of predicting the technical condition, determine potential defects in the units of complex dynamic objects, which will lead to the occurrence of emergencies during their operation.

3. Methods and materials

In [1], a method was developed for identifying potential defects in the components of helicopter turboshaft engines (TE) based on predicting their operational status at flight modes, which makes it possible to use, along with quantitative mathematical models of engines, qualitative and experimental information obtained during flight engine tests, and also, directly, in a helicopter flight. Therefore, this method was chosen in this work as the initial one.

In [1], based on a given data array of the object under research $X(t) = \left(x_{t_1}, x_{t_2}, \dots, x_{t_n}\right)$ input parameters, which correspond only to its operation normal mode, that is, in that time when there were no failures or other anomalies, a neural network model $f(X(t), \theta)$ is developed (θ are the model parameters), based on a fragment with regular values $\left\{x_{t_1}, x_{t_2}, \dots, x_{t_n}\right\}$, which determines whether the fragments $\left\{x_{t_{n+1}}, x_{t_{n+2}}, \dots, x_{t_{n+l}}\right\}$ are anomalous with acceptable accuracy for identifying possible defects of the research object.

The developed method consists of using subsequent data of the initial indicators of the research object to develop a time series predicting model. In this case, the models are trained on data that corresponds only to its operation normal mode, that is, at a time when there were no breakdowns or other anomalies. Thus, the model trains to predict what the signal should be during normal operation. If at a certain point in time, the actual value of the *i*-th initial indicator differs from the predicted normal value, anomalous behaviour is recorded and a signal about a potential defect is generated (Fig. 1).

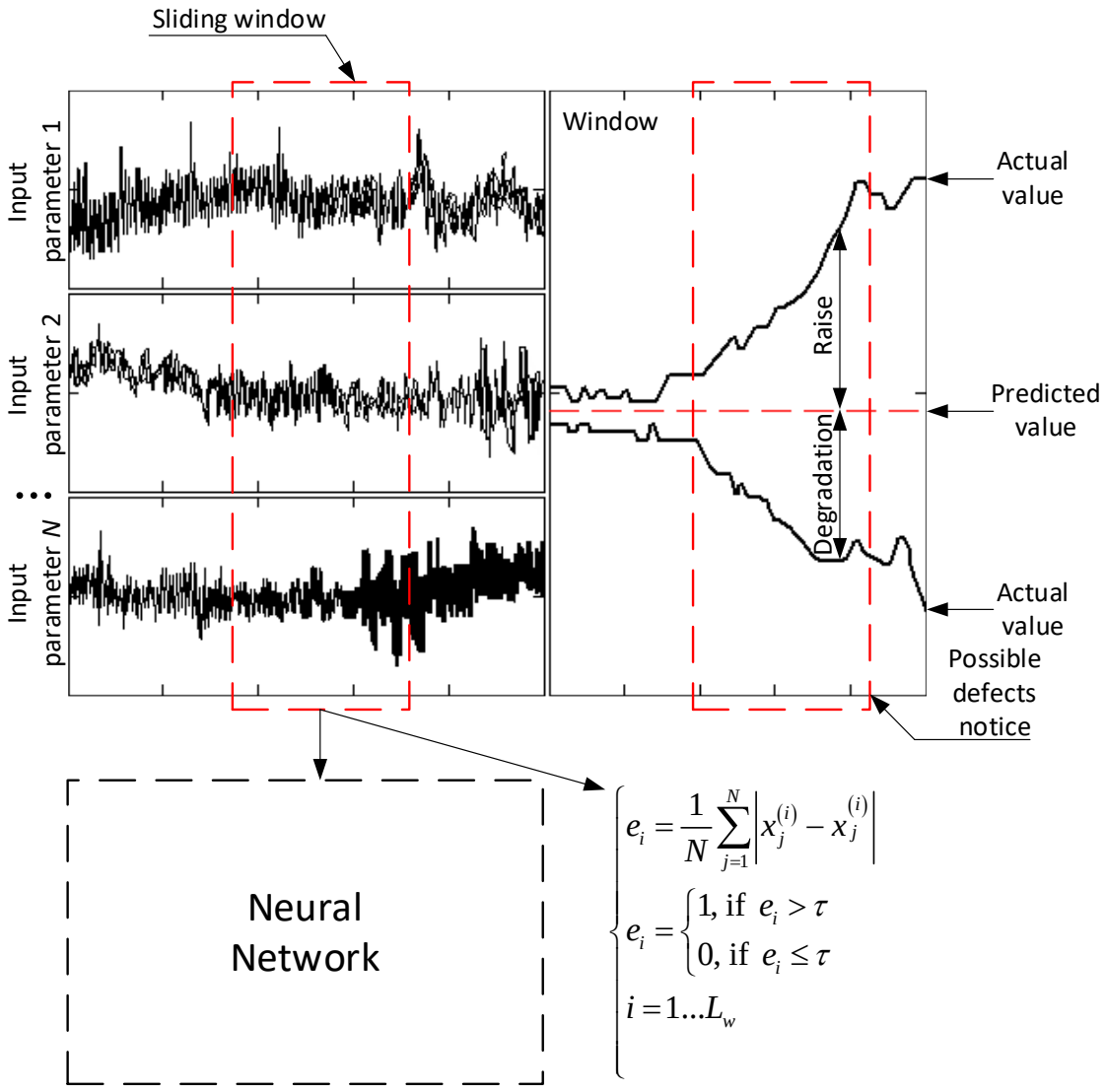


Figure 1: Complex dynamic objects identifying potential defects graphic interpretation. (author's research, generalized based on [1]).

Unlike [1], the proposed neural network method for identifying potential defects in complex dynamic objects is not based on the GRNN neural network, as in [1], but on the Transformer model [21, 22], which is capable of adapting to different lengths of input and output sequences without the need for pre-configuration. At the same time, the Transformer model allows one to take into account dependencies between sequence elements over large time intervals, which is important for predicting defects in complex dynamic objects.

For the normal behaviour of the system, data on the initial indicators of the research object are collected, which must be presented in the form of a time series containing a sequence of actual values. At the same time, instead of a Seq-to-Seq model based on a general regression neural network of the GRNN neural network, it is proposed to use a

model based on deep learning and the Transformer architecture, which is well suited for working with data sequences, allowing to take into account long-term dependencies and work with different lengths of input and output sequences. The proposed Transformer-based model has the following architecture:

1. Encoder: converts the input data sequence into an internal representation, taking into account the context and dependencies between the elements of the sequence. For each element of the input sequence X, the model creates its vector representation x_i, which is then combined into a matrix X_{enc}:

$$X_{enc} = Encoder(X). \tag{1}$$

2. Decoder: generates an output data sequence based on an internal representation, taking into account both the input data and the previous elements of the output sequence. At each step of generating the output sequence Y, the decoder uses the context vectors C from the encoder, as well as the previous elements of the output sequence Y, to predict the next element:

$$Y_t = Decoder(Y_{t-1}, C). \tag{2}$$

3. Positional Encoding: adds vectors representing the positions of elements in a sequence so that the model can take into account the order of the data. For this, sine and cosine functions with different frequencies are used [23, 24]:

$$PE_{pos,2i} = \sin\left(\frac{pos}{10000^{\frac{2\cdot i}{d_{model}}}}\right), \tag{3}$$

$$PE_{pos,2i+1} = \cos\left(\frac{pos}{10000^{\frac{2\cdot i}{d_{model}}}}\right), \tag{4}$$

where pos is the position in the sequence, i is the index of the element in the positional encoding vector, d_{model} is the dimension of the model.

4. Attention Mechanism allows the model to focus on different parts of the input sequence when generating the output sequence [25, 26]. One of the common methods is the scalar product attention mechanism (Scaled Dot-Product Attention) [27, 28]:

$$Attention(Q, K, V) = V \cdot \mathrm{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right), \tag{5}$$

where Q, K, and V are queries, keys and values, d_k is the dimension of the keys. Q queries are usually generated based on the hidden state of the decoder or the last generated element of

the output sequence, and can also be created by linear transformation of the corresponding vectors. Keys *K* are generated from the hidden state of the encoder or a sequence of inputs, similar to queries. The *V* values represent the actual data that the model must rely on when generating the output sequence. The key dimension d_k is a hyperparameter that is empirically selected during model tuning, usually chosen according to the dimension of the model's hidden state and the complexity of the task.

After obtaining predicted values using the Transformer model, the following approach can be used to identify potential defects in the research object:

- If the difference between actual and predicted values exceeds a certain threshold, it may indicate a defect.
- Additionally, statistical measures such as standard deviation can be considered to account for the extent to which forecast values deviate from actual values.

To determine potential defects of the research object, the following expressions are used:

$$Z = \begin{cases} 1, \text{if } A > \rho, \\ 0, \text{if } A \leq \rho, \end{cases} \tag{6}$$

$$A = \frac{1}{L_w} \cdot \sum_{i=1}^{L_w} e_i, \tag{7}$$

$$e_i = \frac{1}{N} \cdot \sum_{i=1}^{N} \left| x_i^{(j)} - \hat{x}_i^{(j)} \right|, \tag{8}$$

$$e_i = \begin{cases} 1, \text{if } e_i > \tau, \\ 0, \text{if } e_i \leq \tau, \end{cases} \tag{9}$$

where *X* is a sequence of actual values of length L_w; *X'* is a sequence of predicted values of features with a length at a point in time; $x_i^{(j)}$ is an actual value of the *i*-th initial indicator at the *j*-th moment; $\hat{x}_i^{(j)}$ is a predicted value of the *i*-th initial indicator at the *j*-th moment.

For a Transformer-based model, it is advisable to use a loss function that reflects the difference between the actual and predicted sequence values [29, 30]. Given that the model generates a sequence, loss functions that estimate the probability of a match between the generated sequence and the actual target sequence are typically used. One of the standard choices for the sequence generation task is cross entropy (categorical cross entropy) [31, 32]. This loss function is widely used in classification and sequence generation problems. The expression describes the cross-entropy loss function between the actual and predicted probability distributions:

$$J = -\sum_{t=1}^{T} y_t \cdot \log(\hat{y}_t), \tag{10}$$

where T is the length of the sequence, y_t is the actual probability distribution for the sequence element at step t, $\hat{y}_t$ is the predicted probability distribution for the sequence element at step t.

The loss function (10) estimates the difference between the actual and predicted probability distributions for each element of the sequence and penalizes the model for incorrect predictions.

It is also worth noting that in the proposed method, the root mean square error can be used to assess the numerical prediction accuracy:

$$MSE = \frac{1}{T} \cdot \sum_{t=1}^{T} (y_t - \hat{y}_t)^2. \tag{11}$$

Currently, Transformer-based models are successfully implemented in the form of graph neural networks, whose ability to train on large amounts of data and effectively work with sequences of any length make them indispensable tools in modern research and development in the field of artificial intelligence [33–35].

Graph Neural Networks (GNN) are a powerful class of neural networks specialized for analyzing data represented as graphs. Unlike GRNN neural networks, GNN has unique advantages such as the ability to take into an account data structure, adaptability to various graph structures, efficiency in modelling complex dependencies, and graph training ability. Therefore, the proposed method uses a graph neural network (Fig. 2) in contrast to [1], where the GRNN neural network was used.

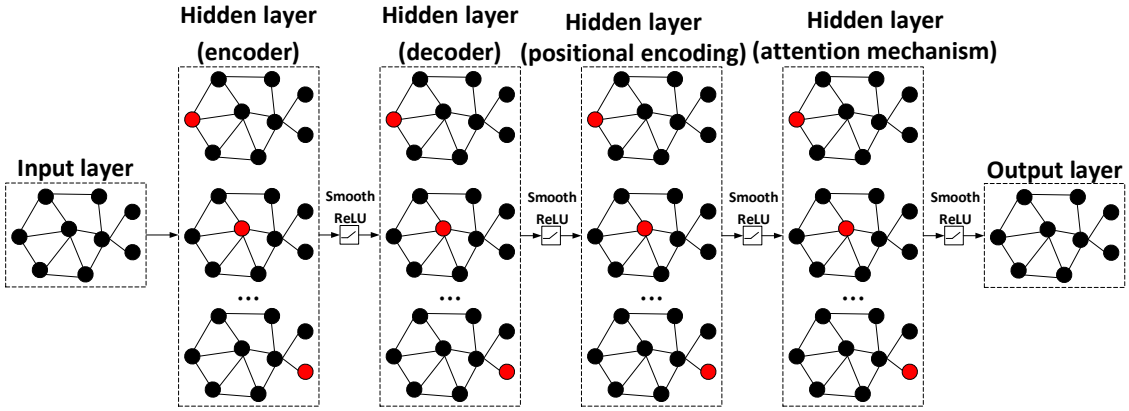


Figure 2: Graph neural network general view. (author's research, based on [33–35]).

At the initial stage of GNN training based on the Transformer architecture, it is assumed that the input data are graphs, where nodes can contain features and edges can have weights or other information. In this case, the graphs are represented in the form of an adjacency matrix or adjacency lists. For the model training also requires dividing the data into training, validation and test sets.

At the next stage, the Transformer architecture is adapted to work with graph data according to (1)–(4). This involves using attention mechanisms (5) to take into account

information about neighbouring nodes and edges in the graph, which involves creating input and output vector representations for nodes and edges and using multilayer perceptron to compute aggregated features.

At the next stage, the loss function is calculated according to (10) and the root mean square error according to (11). To minimize the loss function during training, an optimizer is then selected, for which it is advisable to use Adam [36], which has an adaptive training rate, provides stable training through the use of gradient moments, and usually demonstrates good performance in practice in classification tasks.

In the next stage, the model training process on the training data set is carried out directly, followed by testing its performance on the validation set. For each training epoch, the training set is passed through. Each element of the training set is fed to the input of the model, then the loss function (10) is calculated for this element and the gradients of the loss function are calculated over the model parameters as:

$$\nabla_\theta J(\theta) = \frac{\partial J(\theta)}{\partial \theta}. \tag{12}$$

Using the computed gradients, the model parameters are updated using an update equation based on the Adam optimizer:

$$m = \beta_1 \cdot m + (1 - \beta_1) \cdot \nabla_\theta J(\theta), \tag{13}$$

$$v = \beta_2 \cdot v + (1 - \beta_2) \cdot \left(\nabla_\theta J(\theta)\right)^2, \tag{14}$$

$$\theta = \theta - \alpha_t \cdot \frac{m}{\sqrt{v} + \epsilon}, \tag{15}$$

where β_1 = 0.9...0.999 and β_2 = 0.9...0.999 are exponential smoothing coefficients for estimates of the first and second moments of the gradient, respectively, m and v are estimates of the first and second moments of the gradient, ϵ is a small number for numerical stability, α_t is the adaptive training rate, proposed in this work, calculated according to the expression:

$$\alpha_t = \alpha \cdot \frac{\sqrt{1 - \beta_2^t}}{1 - \beta_1^t}, \tag{16}$$

where α is the specified training rate.

After each training epoch, the model is evaluated on the validation dataset using performance metrics for the given task. For the task of identifying potential defects in complex dynamic objects, it is proposed to use the following performance metrics [37, 38]:

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN}, \tag{17}$$

$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision \cdot Recall},$$

where *Precision* is accuracy, *Recall* is completeness, *TP* (True Positives) is the correctly identified defects number, *FP* (False Positives) is the incorrectly identified defects number, and *FN* (False Negatives) is the defects that the model number could not identify.

If the task of identifying potential defects has several classes of defects, it is proposed to calculate the accuracy for each class of defects to evaluate the performance of the model on each class independently using the *PerClassAccuracy* metric is the percentage of correctly identified defects for each class:

$$PerClassAccuracy = \frac{TP_i}{TP_i + FN_i}, \quad (18)$$

where TP_i (True Positives) is the number of correctly identified defects of the *i*-th class, and FN_i (False Negatives) is the number of incorrectly identified defects of the *i*-th class.

Model training stops when a stopping criterion is reached, for example, the maximum number of epochs, at which if the current training epoch is equal to the maximum number of epochs, then model training is completed. Another stopping criterion is the lack of improvement on the validation dataset for a certain number of epochs. For this criterion, an early stopping mechanism is usually used, which allows you to stop training if the performance of the model on the validation dataset does not improve over a certain number of epochs *P*. Formally, this can be written as follows: let val_L_i be the value of the loss function on the validation dataset data set after the *i*-th training epoch. Then, for the *i*-th training epoch, it is checked whether the model's performance has improved on the validation data set:

$$Impr = val_{L_{i-1}} - val_{L_i}. \quad (19)$$

If an improvement occurs ($Impr > 0$), the epoch counter is reset to zero without improvement (no_improvement_epochs). If there is no improvement, the counter is incremented by 1. If the counter reaches a preset value *P*, then model training stops.

Once the model is trained, its performance on a test dataset is evaluated to evaluate its ability to generalize to new data.

The work proposes to use a new neuron activation function, Smooth ReLU, based on the ReLU function. The main idea of modifying the ReLU function is to make the activation function smoother and more continuous to improve the convergence and robustness of training. The expression describes the Smooth ReLU activation function:

$$f(x) = \begin{cases} x, & \text{if } x > 0, \\ \dfrac{1}{1 + e^{-\gamma \cdot x}}, & \text{if } x \le 0, \end{cases} \quad (20)$$

where γ is a parameter that determines the "degree of smoothness" of the function. When $x > 0$, the function behaves like a regular ReLU. For $x \le 0$, it uses a sigmoid function to

smoothly transition from 0 to negative values. This avoids sudden “steps” in the gradient and can help speed up the neural network's training.

The proposed neural activation function Smooth ReLU retains the benefits of ReLU (no gradient for positive values) and adds smoothness for negative values, which can be useful for improving training in some cases.

4. Experiment

At the preliminary stage of the computational experiment, the feasibility of using the proposed neuron activation function Smooth ReLU (20) is mathematically justified in comparison with the traditional ReLU $f(x) = \max(0, x)$, which is used in graph neural networks. The initial data of the problem is the loss function J (10), which must be minimized, that is, $J \to \min$. To study the activation function of neurons, it is extremely important to analyze their derivatives. The derivative of the activation function allows you to analyze the rate of change of the activation function of neurons depending on changes in input data. This, in turn, helps optimize the process of updating neuron weights during neural network training.

The derivative of the traditional ReLU neuron activation function (Fig. 3, red curve) has the form:

$$f'(x) = \begin{cases} 1, \text{ if } x > 0, \\ 0, \text{ if } x \le 0. \end{cases} \tag{21}$$

The derivative of the proposed Smooth ReLU neuron activation function (Fig. 3, black curve) has the form:

$$f'(x) = \begin{cases} 1, & \text{if } x > 0, \\ \dfrac{\gamma \cdot e^{-\gamma \cdot x}}{(1 + e^{-\gamma \cdot x})^2}, & \text{if } x \le 0. \end{cases} \tag{22}$$

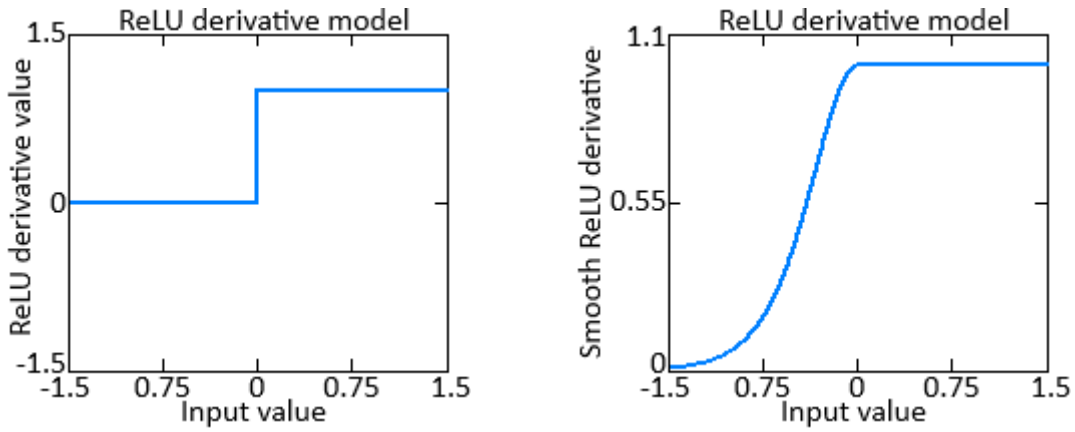


Figure 3: Diagram of the derivative of traditional ReLU function (red curve) and proposed Smooth ReLU function (black curve). (author's research).

As can be seen from (21), (22), as well as from Fig. 3, a, the problem with regular ReLU is that the derivative is 0 for all negative values of x, which can lead to a "dead neurons" problem in a neural network, where neurons stop updating due to a zero gradient. The advantage of Smooth ReLU is that it always has a non-zero gradient for all values of x, including negative values. This avoids the problem of “dead neurons” and promotes more stable training, especially in deep neural networks. Thus, the use of Smooth ReLU (Fig. 3, b) is mathematically justified because it provides a smooth and continuous gradient over the entire definition domain, which can help improve convergence and model training.

Based on [1], to conduct a computational experiment consisting of assessing the performance of the proposed method for identifying potential defects, the research object was TV3-117 is the helicopter TE of various helicopter modifications. To conduct the computational experiment, we used a personal computer with an AMD Ryzen 5 5600 processor, which has 6 cores with 12 threads operating at 3.5 GHz and 32 GB of DDR-4 RAM. At the first stage of the computational experiment, the creation, analysis and pre-processing of a training sample is carried out, consisting of the following parameters that are recorded on board the helicopter (Table 1): n_{TC} is the gas generator rotor r.p.m., n_{FT} is the free turbine rotor speed, T_G is the gas temperature in front of the compressor turbine, h is the flight altitude, T_N is the temperature, P_N is the pressure, ρ is the air density [37–39]. All input parameters are reduced to absolute values according to the theory of gas-dynamic similarity.

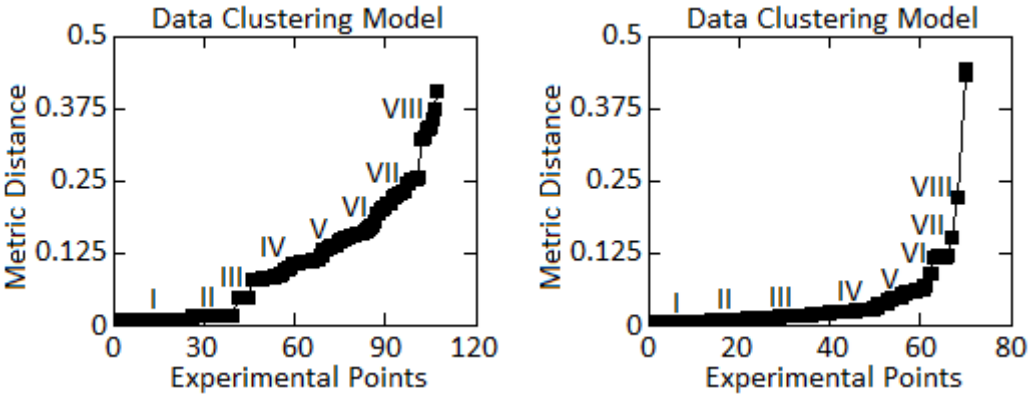
Table 1

The training set part (author's research, published in [37–39])

Number	Gas generator rotor r.p.m.	Free turbine rotor speed	Gas temperature in front of the compressor turbine
1	0.929	0.943	0.932
2	0.933	0.982	0.964
3	0.952	0.962	0.917
4	0.988	0.987	0.908
5	0.991	0.972	0.899
…	…	…	…
256	0.981	0.973	0.953

A detailed description of the process of preprocessing data from the training set (Table 1) is given in [37–39]. This process includes several steps: data homogeneity assessment, dividing them into control and test samples, and checking the representativeness of both samples using cluster analysis. Data homogeneity is assessed using the Fisher-Pearson test [40], the result of which confirms the homogeneity of the samples and the hypothesis of normal distribution (at a significance level of 0.05 and 13 degrees of freedom, the resulting value $\chi^2 = 3.588$ does not exceed the critical value of 3.44). The Fisher-Snedecor test [41] is also used to test homogeneity, confirming the homogeneity of the training sample (at a significance level of 0.01 and 13 degrees of freedom, the resulting value $F = 1.28$ does not exceed the critical value of 22.362). To assess

representativeness, data clusters identified by cluster analysis are examined (Fig. 4). After the randomization procedure, training, and test samples are formed in a 2:1 ratio. Cluster analysis shows that both samples contain eight classes, which indicates their representativeness. The distances between clusters are almost the same in both samples, which confirms their similarity and representativeness.



a b

Figure 4: Cluster analysis results: a – Original experimental dataset (I…VIII – classes); b – Training dataset. (author's research, published in [37–39]).

In the course of graph neural network training (Fig. 2), the proposed algorithm obtained dependences of the accuracy (Fig. 5), completeness (Fig. 6), and losses (Fig. 7) of the neural network on the number of iterations (1000 iterations were used in the work), at which "blue curve" means training on the training set, "orange curve" means validation on the control set. From Fig. 5 it can be seen that the limiting value of accuracy practically reaches 1, Fig. 6 it is clear that the completeness is almost constant and equal to 1, and Fig. 7 shows that the loss value does not exceed 0.025, which indicates the high efficiency of training the model on the available data and its ability to accurately generalize to new data.

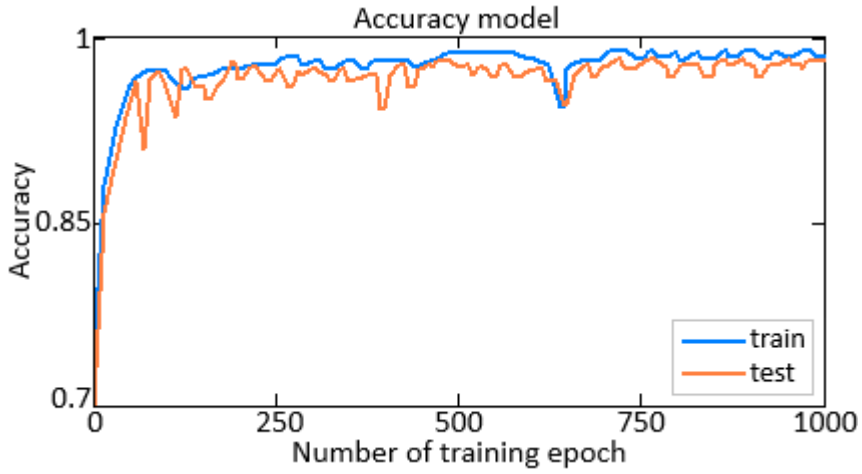


Figure 5: Diagram of changes in the neural network accuracy function with 1000 iterations. (author's research).

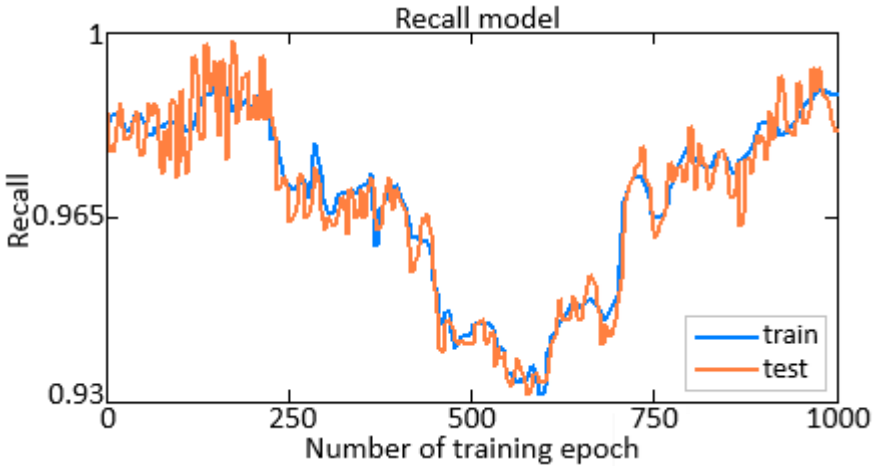


Figure 6: Diagram of changes in the neural network recall function with 1000 iterations. (author's research).

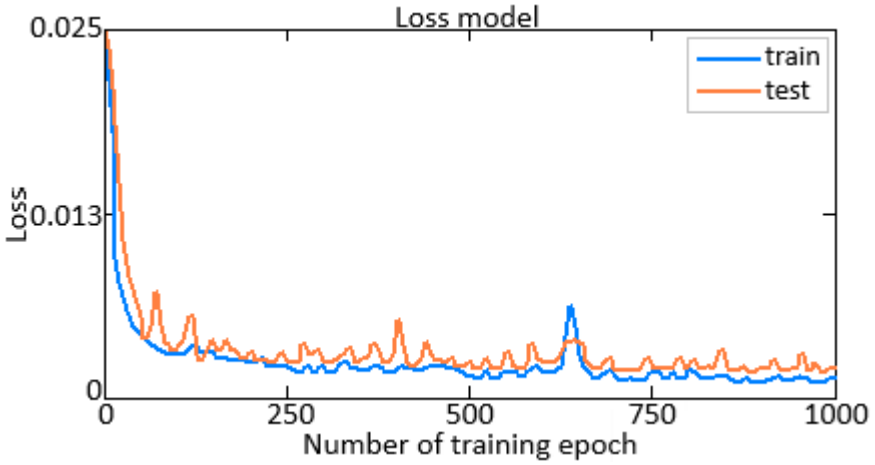


Figure 7: Diagram of changes in the neural network loss function with 1000 iterations. (author's research).

Such results make the model potentially suitable for solving the task of identifying potential defects in helicopter TE.

5. Results

Fig. 8 shows the results of identifying potential defects based on long-term prediction of the gas temperature parameter in front of the compressor turbine T_G in the components of the TV3-117 TE. On the graph of the initial parameter T_G, corresponding to the time point t = 94.65 hours, predicting is carried out. At t = 96.2 hours, there is a decrease in the T_G parameter by 4 %, and at t = 99.3 hours – by another 3%, for a total of 7 % decrease. At times t = 96.2 hours and t = 99.3 hours Fig. 9 and 10 show the corresponding bursts, indicating a possible malfunction of the engine combustion chamber – the potential appearance of cracks (burnouts). In this way, normal engine operation can be guaranteed

for 8 hours. The results of these researches fully confirm previously conducted identical research in [1].

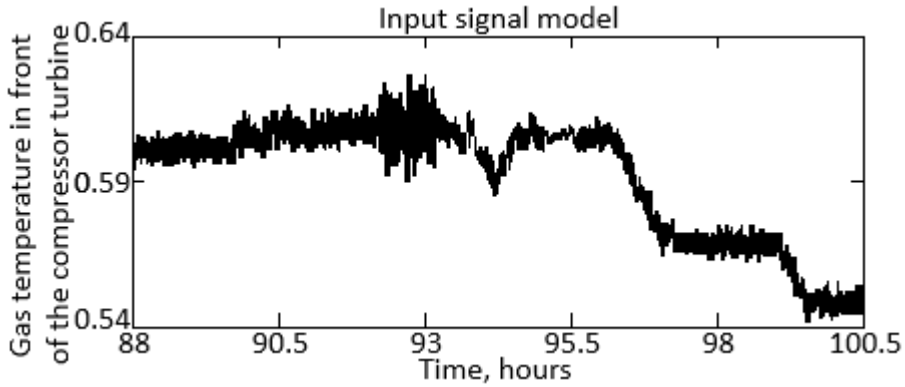


Figure 8: The results of identification of potential defects in the combustion chamber of TV3-117 TE. (author's research).

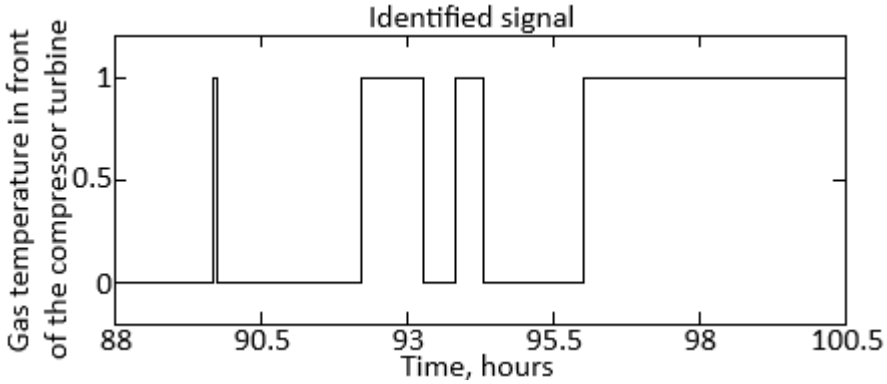


Figure 9: The results of identification of potential defects in the combustion chamber of TV3-117 TE. (author's research).

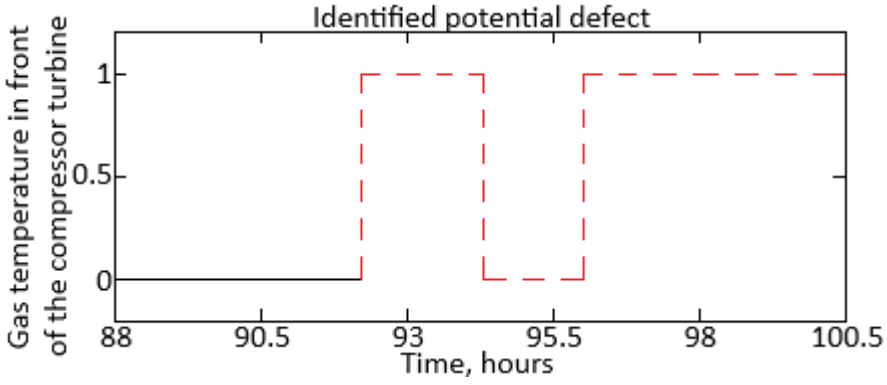


Figure 10: The results of identification of potential defects in the combustion chamber of TV3-117 TE. (author's research).

The red curve in Fig. 10 means an identified probable defect – the potential appearance of cracks (burnouts) in the combustion chamber. The obtained data on long-term prediction of the gas temperature parameter in front of the compressor turbine T_G allows us to take effective measures to detect and prevent potential defects in the engine combustion chamber, which in turn helps to improve the safety and reliability of helicopter TE. Thus, these researches have significant practical significance and can be used in the field of maintenance and aviation equipment operation.

6. Discussion

Similar to [1], as a result of the comparative experiment, the results were obtained (Table 2) of the developed and known methods: Simple Autoencoder, Autoencoder using the LSTM neural network, the method developed in [1] using the GRNN neural network and the developed method using graph neural network.

Table 2
Comparative experiment results (author's research)

Parameter	SOTA-architectures						Developed in [1] method using the GRNN neural network			Developed method using the graph neural network		
	Simple AutoEncoder			AutoEncoder using the LSTM neural network								
	Precision	Recall	F1-scope	Precision	Recall	F1-scope	Precision	Recall	F1-scope	Precision	Recall	F1-scope
$\mathbf{n_{TC}}$	0.81	0.76	0.78	0.63	0.67	0.64	0.88	0.87	0.87	0.96	0.98	0.94
$\mathbf{T_G}$	0.86	0.84	0.85	0.77	0.77	0.77	0.92	0.91	0.91	0.99	0.97	0.96
$\mathbf{n_{FT}}$	0.61	0.57	0.57	0.67	0.64	0.67	0.79	0.78	0.78	0.95	1.0	0.93

Table 2 shows that the results of the proposed method using a graph neural network are superior to almost all presented accuracy metrics. Alternative methods (Simple AutoEncoder [42, 43], LSTM AutoEncoder [44, 45], and a similar method using the GRNN neural network developed in [1]) also demonstrate deterioration in performance on the same signals. Table 2 also shows that the results for all *Precision*, *Recall*, and F1-scope metrics when using a graph neural network are better compared to Simple AutoEncoder, LSTM AutoEncoder, and a similar method using the GRNN neural network [1]. These results confirm the effectiveness and high accuracy of the proposed method using a graph neural network while achieving high accuracy for all analyzed transformations. Thus, the results of the proposed method are superior to almost all presented accuracy metrics compared to alternative methods: Simple AutoEncoder, LSTM AutoEncoder, and a similar method using the GRNN neural network [1].

At the final stage of the comparative experiment, the effectiveness of the proposed method was assessed, which consisted of conducting a comparative analysis of the accuracy

of classical and neural network methods for identifying potential defects, the results of which are given in Table 3, which displays the results of calculating the probability of errors of I and II types when identifying defects in the compressor (n_{TC} parameter), combustion chamber (T_G parameter) and compressor turbine (n_{FT} parameter). In this case, errors of the first (*FPR*) and second (*FNR*) types were calculated as [46, 47]:

$$FPR = \frac{FP}{FP + TN}, FNR = \frac{FN}{FN + TP}, \tag{23}$$

where False Positives (*FP*) is the number of observations that are incorrectly classified as positive, False Negatives (*FN*) is the number of observations that are incorrectly classified as negative, True Positives (*TP*) is the number of observations that are correctly classified as positive, True Negatives (*TN*) is the number of observations that are correctly classified as negative.

A type I error means rejecting the null hypothesis when it is true, while a type II error means accepting the null hypothesis when it is false. In the context of identifying potential defects, the null hypothesis is defined as: "There are no potential defects in the test item." This means that under the null hypothesis, there are no anomalies, errors, or defects to be identified or corrected.

Table 3
Results of calculating errors of the I and II types (author's research)

Identification Method	**Probability of error in potential defect identification, %**					
	Compressor defect		**Combustion chamber defect**		**Compressor turbine defect**	
	Type I error	**Type II error**	**Type I error**	**Type II error**	**Type I error**	**Type II error**
Classical method (Tolerance control)	1.75	1.35	2.48	1.76	2.23	1.51
Neural Network Method:						
Simple AutoEncoder	1.15	0.94	1.24	1.09	1.23	1.06
LSTM AutoEncoder	0.95	0.58	1.02	0.63	0.99	0.61
GRNN neural network [1]	0.62	0.31	0.61	0.29	0.60	0.27
Graph neural network	0.33	0.17	0.33	0.16	0.32	0.14

As can be seen from Table 3, the use of a graph neural network reduces errors of the first and second types compared to the use of Simple AutoEncoder by 3.49...7.58 times, LSTM AutoEncoder – by 2.88...4.36 times, and the GRNN neural network [1] – by 1.81...1.93 times. Therefore, we can conclude that the use of a graph neural network in the method of identifying potential defects reduces errors of the first and second types by almost 2 times compared with the use of the GRNN neural network [1].

For four classification classes (True Positives, True Negatives, False Positives, False Negatives) confusion matrix was developed (Table 4) [48–50]. Each cell of the confusion matrix shows the number of times the actual class (rows) was classified as the predicted class (columns) for each method.

Table 4
Developed confusion matrix (author's research)

Actual \ Predicted	Developed method using the graph neural network	Developed in [1] method using the GRNN neural network	Simple AutoEncoder	AutoEncoder using the LSTM neural network
True Positives	95	3	2	0
True Negatives	2	90	6	2
False Positives	1	5	85	9
False Negatives	0	2	7	91

The confusion matrix demonstrates that the GNN is the most accurate method, correctly classifying the majority of instances in all classes with minimal misclassifications. The GRNN network, while performing well, shows slightly more errors, particularly in distinguishing between classes “True Negatives” and “False Positives”. The Simple Autoencoder demonstrates moderate accuracy but has noticeable misclassifications across all classes, particularly in misidentifying class “False Positives” instances. The Autoencoder using the LSTM neural network exhibits the lowest accuracy, with significant misclassifications, especially confusing instances of classes “True Positives”, “True Negatives” and “False Positives”, while showing better performance for class “False Negatives”. Overall, the GNN method clearly outperforms the others, followed by GRNN, Simple Autoencoder, and LSTM Autoencoder in descending order of classification quality.

To conduct ROC analysis for four methods (Graph Neural Network, GRNN Network, Simple Autoencoder, Autoencoder using the LSTM neural network), true positive and false positive rates were calculated for each class and method, and then the corresponding ROC curves were plotted. To do this, a binary classification is created for each class (this class versus all others). For each class, True Positive Rate (TPR) and False Positive Rate (FPR) are calculated as [49–51]:

$$TPR = \frac{TP}{FP + TN}, FPR = \frac{FP}{FP + TN}. \quad (24)$$

The area under the ROC curve (AUC) can be estimated using the trapezium equation. If (TPR_i and FPR_i) are the ROC curve points coordinates, then AUC can be calculated as follows:

$$AUC = \sum_{i=1}^{P-1} \frac{TPR_i + TPR_{i+1}}{2} \cdot (FPR_{i+1} - FPR_i), \quad (25)$$

where *P* is the points number on the ROC curve. Table 5 shows the ROC analysis results.

Table 5
ROC analysis results (author's research)

Actual \ Predicted	Developed method using the graph neural network	Developed in [1] method using the GRNN neural network	Simple AutoEncoder	AutoEncoder using the LSTM neural network
True Positives	95	90	2	0
True Negatives	3	7	8	10
False Positives	270	266	290	290
False Negatives	30	35	98	100
TPR	0.76	0.72	0.02	0
FPR	0.011	0.025	0.027	0.33
AUC	0.838	0.724	0.445	0.292

Thus, the use of GNN gives high accuracy with a low level of false positive results; the use of GNN also gives high accuracy, but 1.16 times lower than that of GNN. Using Simple Autoencoder gives moderate accuracy, with noticeable errors. Using LSTM Autoencoder gives the lowest accuracy with the highest number of false positives.

7. Conclusions

The neural network method for identifying potential defects in complex dynamic objects (using the example of helicopter turboshaft engines) has been further developed, which differs from the existing one in that, through the use of the Transformer model rather than the GRNN neural network, it has made it possible to identify potential defects in complex dynamic objects with almost 100 % accuracy (using the example of the potential appearance of cracks (burnouts) in the combustion chamber of helicopter turboshaft engines due to a predicted decrease in gas temperature in front of the compressor turbine values).

A neural network implementation of the Transformer model is proposed using a graph neural network in the proposed method for identifying potential defects in complex dynamic objects, which made it possible to reduce errors of the first and second types in comparison with the use of Simple AutoEncoder by 3.49...7.58 times, LSTM AutoEncoder by 2.88...4.36 times, neural GRNN network – 1.81…1.93 times.

Improved ReLU activation function in the form of Smooth ReLU to make the activation function smoother and more continuous, and, as a result, improve the convergence and stability of training. By analyzing the derivatives of the ReLU and Smooth ReLU functions, it is determined that the Smooth ReLU function solves the "dead neurons" task associated with regular ReLU by providing a non-zero gradient for all input data values, including negative ones, which provides more stable training of neural networks and prevents neurons from stopping updating due to zero gradient.

Acknowledgements

This research was funded by the Ministry of Internal Affairs of Ukraine "Theoretical and applied aspects of the development of the aviation sphere" under Project No. 0123U104884.

References

[1] S. Vladov, Y. Shmelov, R. Yakovliev, Modified Method of Identification Potential Defects in Helicopters Turboshaft Engines Units Based on Prediction its Operational Status, in: Proceedings of the 2022 IEEE 4th International Conference on Modern Electrical and Energy System (MEES), Kremenchuk, Ukraine, October 20–22, 2022, pp. 556–561. doi: 10.1109/MEES58014.2022.10005605

[2] O. Ivanov, L. Koretska, V. Lytvynenko, Intelligent modeling of unified communications systems using artificial neural networks, CEUR Workshop Proceedings 2623 (2020) 77–84.

[3] S. Babichev, V. Lytvynenko, J. Skvor, J. Fiser, Model of the objective clustering inductive technology of gene expression profiles based on SOTA and DBSCAN clustering algorithms, Advances in Intelligent Systems and Computing 689 (2018) 21–39. doi: 10.1007/978-3-319-70581-1_2

[4] J. Chen, Y. Liu, Neural optimization machine: a neural network approach for optimization and its application in additive manufacturing with physics-guided learning, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 381: 2260 (2023). doi: 10.1098/rsta.2022.0405. URL: https://royalsocietypublishing.org/doi/10.1098/rsta.2022.0405

[5] W.-K. Hong, 4 - Forward and backpropagation for artificial neural networks, in: W.-K. Hong (Ed.), Artificial Intelligence-Based Design of Reinforced Concrete Structures, Woodhead Publishing, Sawston, England, 2023, pp. 67–116. doi: 10.1016/B978-0-443-15252-8.00006-6.

[6] M. Heidari, M. H. Moattar, H. Ghaffari, Forward propagation dropout in deep neural networks using Jensen–Shannon and random forest feature importance ranking, Neural Networks 165 (2023) 238–247. doi: 10.1016/j.neunet.2023.05.044.

[7] A. Kupina, D. Zubov, Y. Osadchuka, R. Ivchenkoa, V. Saiapin, Intelligent Neural Networks Models for the Technological Separation Processes, CEUR Workshop Proceedings 3373 (2023) 76–86.

[8] B. Rusyn, R. Kosarevych, O. Lutsyk, V. Korniy, Segmentation of atmospheric cloud images obtained by remote sensing, in: Proceedings of the 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, February 20–24, 2018, pp. 213–216. doi: 10.1109/TCSET.2018.8336189

[9] S. Babichev, M. Korobchynskyi, O. Lahodynskyi, O. Korchomnyi, V. Basanets, V. Borynskyi, Development of a technique for the reconstruction and validation of gene network models based on gene expression, Eastern-European Journal of Enterprise Technologies 1(4 (91) (2018) 19–32. doi: 10.15587/1729-4061.2018.123634

[10] B. Li, Y.-P. Zhao, Y.-B. Chen, Unilateral alignment transfer neural network for fault diagnosis of aircraft engine, Aerospace Science and Technology, vol. 118 (2021) 107031. doi: 10.1016/j.ast.2021.107031.
[11] M. Soleimani, F. Campean, D. Neagu, Diagnostics and prognostics for complex systems: A review of methods and challenges, Quality and Reliability Engineering, vol. 37, issue 8 (2021) 3746–3778 doi: 10.1002/qre.2947.
[12] X. Zhu, M. Li, X. Liu, Y. Zhang, A backpropagation neural network-based hybrid energy recognition and management system, Energy 297 (2024) 131264. doi: 10.1016/j.energy.2024.131264
[13] Z. Wang, B. Li, W. Li, S. Niu, M. Wang, T. Niu, NAS-ASDet: An adaptive design method for surface defect detection network using neural architecture search, Advanced Engineering Informatics 61 (2024) 102500. doi: 10.1016/j.aei.2024.102500
[14] B. Chen, T. Niu, R. Zhang, H. Zhang, Y. Lin, B. Li, Feature matching driven background generalization neural networks for surface defect segmentation, Knowledge-Based Systems 287 (2024) 111451. doi: 10.1016/j.knosys.2024.111451
[15] S. Vladov, Y. Shmelov, R. Yakovliev, Helicopters Aircraft Engines Self-Organizing Neural Network Automatic Control System, CEUR Workshop Proceedings 3137 (2022) 28–47. doi: 10.32782/cmis/3137-3
[16] Y. Shen, K. Khorasani, Hybrid multi-mode machine learning-based fault diagnosis strategies with application to aircraft gas turbine engines, Neural Networks, vol. 130 (2020) 126–142. doi: 10.1016/j.neunet.2020.07.001
[17] L. Xu, S. Dong, H. Wei, D. Peng, W. Qian, Q. Ren, L. Wang, Y. Ma, Intelligent identification of girth welds defects in pipelines using neural networks with attention modules 127 (part B) (2024) 107295. doi: 10.1016/j.engappai.2023.107295
[18] J. Xuan, Z. Wang, S. Li, A. Gao, C. Wang, T. Shi, Measuring compound defect of bearing by wavelet gradient integrated spiking neural network, Measurement 223 (2023) 113796. doi: 10.1016/j.measurement.2023.113796
[19] R. Kosarevych, O. Lutsyk, B. Rusyn, Detection of pixels corrupted by impulse noise using random point patterns, The Visual Computer: International Journal of Computer Graphics 38(11) (2022) 3719–3730. doi: 10.1007/s00371-021-02207-1
[20] B. Rusyn, O. Lutsyk, R. Kosarevych, T. Maksymyuk, J. Gazda, Features extraction from multi-spectral remote sensing images based on multi-threshold binarization, Scientific Reports 13(1) (2023) 19655. doi: 10.1038/s41598-023-46785-7
[21] L. Feng, A. Sinchai, Transfer learning model for cash-instrument prediction adopting a Transformer derivative, Journal of King Saud University - Computer and Information Sciences 36:3 (2024) 102000. doi: 10.1016/j.jksuci.2024.102000
[22] Z. Xin, S. Sirejiding, Y. Lu, Y. Ding, C. Wang, T. Alsarhan, H. Lu, TFUT: Task fusion upward transformer model for multi-task learning on dense prediction, Computer Vision and Image Understanding 244 (2024) 104014. doi: 10.1016/j.cviu.2024.104014
[23] X. Wang, S. Chen, L. Chen, D. Zhu, Y. Liu, T. Wu, A hybrid MLP-CNN model based on positional encoding for daytime radiative cooler, Optics Communications 560 (2024) 130448. doi: 10.1016/j.optcom.2024.130448

[24] P. Liu, L. Chen, H. Zhang, Y. Zhang, C. Liu, C. Li, Z. Wang, PEAR: Positional-encoded Asynchronous Autoregression for satellite anomaly detection, Pattern Recognition Letters 176 (2023) 96–101. doi: 10.1016/j.patrec.2023.10.007
[25] X. Wang, W. Deng, Z. Meng, D. Chen, Hybrid-attention mechanism based heterogeneous graph representation learning, Expert Systems with Applications 250 (2024) 123963. doi: 10.1016/j.eswa.2024.123963
[26] M. Han, L. Fan, A short-term energy consumption forecasting method for attention mechanisms based on spatio-temporal deep learning, Computers and Electrical Engineering 114 (2024) 109063. doi: 10.1016/j.compeleceng.2023.109063
[27] T. Xia, X. Chen, Category-learning attention mechanism for short text filtering, Neurocomputing 510 (2022) 15–23. doi: 10.1016/j.neucom.2022.08.076
[28] R. Khalitov, T. Yu, L. Cheng, Z. Yang, Sparse factorization of square matrices with application to neural attention modeling, Neural Networks 152 (2022) 160–168. doi: 10.1016/j.neunet.2022.04.014
[29] M. Zeynali, H. Seyedarabi, R. Afrouzian, Classification of EEG signals using Transformer based deep learning and ensemble models, Biomedical Signal Processing and Control 86 (part A) (2023) 105130. doi: 10.1016/j.bspc.2023.105130
[30] X. Li, Q.-L. Sun, Y. Zhang, J. Sha, M. Zhang, Environmental Modelling & Software 177 (2024) 106042. doi: 10.1016/j.envsoft.2024.106042
[31] J. Chan, I. Papaioannou, D. Straub, Bayesian improved cross entropy method for network reliability assessment, Structural Safety 103 (2023) 102344. doi: 10.1016/j.strusafe.2023.102344
[32] B. Liu, H. Chen, K. Li, M. Ying Yang, Transformer-based multimodal change detection with multitask consistency constraints, Information Fusion 108 (2024) 102358. doi: 10.1016/j.inffus.2024.102358
[33] P. Foroutan, S. Lahmiri, Deep learning-based spatial-temporal graph neural networks for price movement classification in crude oil and precious metal markets, Machine Learning with Applications 16 (2024) 100552. doi: 10.1016/j.mlwa.2024.100552
[34] W. Pei, W.N. Xu, Z. Wu, W. Li, J. Wang, G. Lu, X. Wang, Saliency-aware regularized graph neural network, Artificial Intelligence 328 (2024) 104078. doi: 10.1016/j.artint.2024.104078
[35] Y. Li, C. Jian, G. Zang, C. Song, X. Yuan, Node classification oriented Adaptive Multichannel Heterogeneous Graph Neural Network, Knowledge-Based Systems 292 (2024) 111618. doi: 10.1016/j.knosys.2024.111618
[36] J. Li, Y. Song, X. Song, D. Wipf, On the Initialization of Graph Neural Networks. in: Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA, 2023, pp. 19911–19931. doi: 10.48550/arXiv.2312.02622
[37] S. Vladov, R. Yakovliev, O. Hubachov, J. Rud, Neuro-Fuzzy System for Detection Fuel Consumption of Helicopters Turboshaft Engines, CEUR Workshop Proceedings 3628 (2024) 55–72.
[38] S. Vladov, R. Yakovliev, O. Hubachov, J. Rud, Y. Stushchanskyi, Neural Network Modeling of Helicopters Turboshaft Engines at Flight Modes Using an Approach Based on “Black Box” Models, CEUR Workshop Proceedings 3624 (2024) 116–135.

[39] S. Vladov, Y. Shmelov, R. Yakovliev, M. Petchenko, Neural Network Method for Parametric Adaptation Helicopters Turboshaft Engines On-Board Automatic Control, CEUR Workshop Proceedings 3403 (2023) 179–195.
[40] F. S. Corotto, Appendix C - The method attributed to Neyman and Pearson, Wise Use of Null Hypothesis Tests (2023) 179–188. doi: 10.1016/B978-0-323-95284-2.00012-4
[41] F. V. Motsnyi, Analysis of Nonparametric and Parametric Criteria for Statistical Hypotheses Testing. Chapter 1. Agreement Criteria of Pearson and Kolmogorov, Statistics of Ukraine 4'2018 (83) (2018) 14–24. doi: 10.31767/su.4(83)2018.04.02
[42] S. Haidong, J. Hongkai, Z. Huiwei, W. Fuan, A novel deep autoencoder feature learning method for rotating machinery fault diagnosis, Mechanical Systems and Signal Processing, 95 (2017) 187–204. doi: 10.1016/j.ymssp.2017.03.034
[43] A. Li, C. Feng, S. Xu, Y. Cheng, Graph t-SNE multi-view autoencoder for joint clustering and completion of incomplete multi-view data, Knowledge-Based Systems 284 (2024) 111323. doi: 10.1016/j.knosys.2023.111324
[44] X. Tong, J. Wang, C. Zhang, T. Wu, H. Wang, Y. Wang, LS-LSTM-AE: Power load forecasting via Long-Short series features and LSTM-Autoencoder, Energy Reports, 8:1 (2022) 596–603. doi: 10.1016/j.egyr.2021.11.172
[45] L. Zeng, Q. Jin, Z. Lin, C. Zheng, Y. Wu, X. Wu, X. Gao, Dual-attention LSTM autoencoder for fault detection in industrial complex dynamic processes, Process Safety and Environmental Protection 185 (2024) 1145–1159. doi: 10.1016/j.psep.2024.02.042
[46] S. Vladov, Y. Shmelov, R. Yakovliev, M. Petchenko, S. Drozdova, Neural Network Method for Helicopters Turboshaft Engines Working Process Parameters Identification at Flight Modes, in: Proceedings of the 2022 IEEE 4th International Conference on Modern Electrical and Energy System (MEES), Kremenchuk, Ukraine, 2022, pp. 604–609. doi: 10.1109/MEES58014.2022.10005670
[47] S. Vladov, Y. Shmelov, R. Yakovliev, M. Petchenko, S. Drozdova, Helicopters Turboshaft Engines Parameters Identification at Flight Modes Using Neural Networks, in: Proceedings of the IEEE 17th International Conference on Computer Science and Information Technologies (CSIT), Lviv, Ukraine, 2022, pp. 5–8. doi: 10.1109/CSIT56902.2022.10000444
[48] K. Andriushchenko, V. Rudyk, O. Riabchenko, M. Kachynska, N. Marynenko, L. Shergina, V. Kovtun, M. Tepliuk, A. Zhemba, O. Kuchai. Processes of managing information infrastructure of a digital enterprise in the framework of the «Industry 4.0» concept, Eastern-European Journal of Enterprise Technologies 1(3–97) (2019) 60–72. doi: 10.15587/1729-4061.2019.157765
[49] V. V. Morozov, O. V. Kalnichenko, O. O. Mezentseva, The method of interaction modeling on basis of deep learning the neural networks in complex it-projects, International Journal of Computing 19(1) (2020) 88–96. doi: 10.47839/ijc.19.1.1697
[50] S. Bezobrazov, V. Golovko, A. Sachenko, M. Komar, R. Dolny, V. Kasyanik, P. Bykovyy, E. Mikhno, O. Osolinskyi, Deep multilayer neural network for predicting the winner of football matches, International Journal of Computing 19(1) (2020) 70–77. doi: 10.47839/ijc.19.1.1695
[51] S. Vladov, R. Yakovliev, M. Bulakh, V. Vysotska. Neural Network Approximation of Helicopter Turboshaft Engine Parameters for Improved Efficiency, Energies 17(9) (2024) 2233. doi: 10.3390/en17092233