MACHINE BUILDING
МАШИНОСТРОЕНИЕ

UDC 621.86/.87
https://doi.org/10.23947/2541-9129-2024-8-2-68-77

Research Article

EDN: BMJTNE

Comparative Analysis of the Performance of Artificial Neural Networks in Assessing the Technical Condition of Steel Ropes

Roman V. Khvan
Don State Technical University, Rostov-on-Don, Russian Federation
✉ khvanroman@yandex.ru

Abstract
Introduction. Currently, artificial neural networks (ANN) are successfully used for technical diagnostics of steel ropes. Expensive software products with an adapted neural network implementation environment, such as STATISTICA, Amygdala, MatLab Simulink, are often used for this purpose. The most affordable way to build and train an ANN, from a financial point of view, is to write your own program code using interactive libraries such as TensorFlow, PyTorch, Scikit-learn. However, such libraries are not fully adapted for building an ANN, and to use them you need to have basic programming skills. As a result, the quality of an ANN depends not only on its architecture, training data, and composition, but also on the environment in which it is built. The aim of the work was to compare the quality of the ANN, built and trained by various methods according to the criterion of test network performance, confidence levels for assessing the technical condition of the rope, as well as the complexity and speed of training. For this purpose, new software has been developed to solve the problem of assessing the technical condition of a steel rope using a combination of various rejection indicators.
Materials and Methods. The basis for an ANN training was a statistical database of typical damages of steel ropes and, an expert assessment of the technical condition of steel ropes. The software was written in the Python programming language. Various methods of programming a neural network were presented: an ANN built on the basis of the STATISTICA software package and an ANN built using the interactive Scikit-learn library. Ten test samples were prepared to verify the operation of the ANN. The ANN quality was assessed based on the test network performance and confidence probabilities (activation levels of the "winning" neuron) of determining the technical condition of the rope.
Results. The construction of the ANN using the interactive library Scikit-learn showed a relatively high complexity of construction and a relatively low learning rate of the ANN. Test performance of the network, with a test sample size of ten, turned out to be the same for both built ANNs. At the same time, there was a difference in the indicator of the average confidence level for determining the technical condition of a steel rope between the results of the ANN built on the basis of the STATISTICA software package and the ANN built using the Scikit-learn interactive library.
Discussion and Conclusion. The results showed that the ANN built using the STATISTICA software package with the same architecture and network learning parameters had more optimal software algorithms according to the criteria of confidence probability and network learning speed in comparison with the ANN built using the free Skicit-learn library. However, the indicator of the ANN test performance turned out to be the same for both ANNs. This result justified the use of TensorFlow, PyTorch, and Skicit-learn libraries by the world's leading research and commercial centers in the field of artificial intelligence. The obtained scientific result allows us to numerically evaluate and compare the quality of an ANN having the same architecture and learning parameters, but built using different methods. This will be useful for future scientific research in the field and for selecting the optimal environment for constructing ANNs in industrial applications.

Keywords: steel rope, artificial neural networks, technical condition assessment, Python, Skicit-learn, STATISTICA, rejection indicators

Acknowledgements. The author would like to expresses their gratitude to the team at the Engineering and Consulting Center “Mysl' LLC” of the Novocherkassk NSTU for the opportunity to use the data on the operation of steel ropes, specifically the statistical database on typical damages to steel ropes.

For Citation. Khvan RV. Comparative Analysis of the Performance of Artificial Neural Networks in Assessing the Technical Condition of Steel Ropes. *Safety of Technogenic and Natural Systems.* 2024;8(2):68–77. https://doi.org/10.23947/2541-9129-2024-8-2-68-77

Научная статья

Сравнительный анализ качества работы искусственных нейронных сетей для оценки технического состояния стального каната

Р.В. Хван
Донской государственный технический университет, г. Ростов-на-Дону, Российская Федерация
✉ khvanroman@yandex.ru

Аннотация
Введение. В настоящее время искусственные нейронные сети (ИНС) успешно применяются для технического диагностирования стальных канатов. Зачастую при этом используют дорогостоящие программные продукты с адаптированной средой реализации нейронных сетей, такие как STATISTICA, Amygdala, MATLAB Simulink. Наиболее доступным способом построения и обучения ИНС с финансовой точки зрения является написание собственного программного кода с использованием интерактивных библиотек, таких как TensorFlow, PyTorch, Scikit-learn. Однако такие библиотеки не являются полноценными адаптированными средами построения ИНС, и для их использования необходимо владеть первичными навыками программирования. Поэтому качество ИНС зависит не только от архитектуры, объема и состава обучающих выборок, но и от метода (среды) построения ИНС. Целью данного исследования является сравнение качества работы ИНС, построенных и обученных различными методами, по критерию тестовой производительности сети, доверительным уровням оценки технического состояния каната, а также трудоемкости и скорости обучения. В связи с этим создано новое программное обеспечение для решения задачи оценки технического состояния стального каната по комбинации различных браковочных показателей.
Материалы и методы. Основой для обучения ИНС послужили статистическая база данных типовых повреждений стальных канатов, экспертная оценка их технического состояния. Программное обеспечение написано на языке программирования Python. Приведены различные методы программирования нейронной сети: ИНС, построенной на базе программного комплекса STATISTICA, и ИНС, построенной с использованием интерактивной библиотеки Scikit-learn. Для проверки работы ИНС было подготовлено 10 тестовых выборок. Оценка качества работы ИНС проводилась по тестовой производительности сети и доверительным вероятностям (уровням активации «победившего» нейрона) определения технического состояния каната.
Результаты исследования. Построение ИНС с использованием интерактивной библиотеки Scikit-learn показало сравнительно большую трудоемкость построения и сравнительно небольшую скорость обучения. Тестовая производительность сети при объеме тестовой выборки 10 оказалась одинаковой для обеих построенных ИНС. При этом обнаружилась разница в показателе среднего доверительного уровня определения технического состояния стального каната по результатам работы ИНС, построенной на базе программного комплекса STATISTICA, и ИНС, построенной с использованием интерактивной библиотеки Scikit-learn.
Обсуждение и заключение. Полученные результаты показали, что ИНС, построенная с использованием программного комплекса STATISTICA, при одинаковой архитектуре и параметрах обучения сети имеет более оптимальные программные алгоритмы по критериям доверительной вероятности и скорости обучения сети по сравнению с ИНС, построенной с использованием бесплатной библиотеки Skicit-learn. Однако показатель тестовой производительности ИНС оказался одинаковым для обеих ИНС. Такой результат обосновывает использование ведущими мировыми научно-исследовательскими и коммерческими центрами в области искусственного интеллекта библиотек TensorFlow, PyTorch, Scikit-learn. Полученный научный результат позволит численно оценить и сравнить качество искусственных нейронных сетей, имеющих одинаковые архитектуру и параметры обучения, но построенных различными методами, он будет полезным как для будущих научных исследований в этой области, так и для выбора оптимальной среды построения ИНС в промышленной сфере деятельности.

Ключевые слова: стальной канат, искусственные нейронные сети, оценка технического состояния, Python, Scikit-learn, STATISTICA, браковочные показатели

Благодарности. Автор выражает благодарность коллективу ООО «ИКЦ «Мысль»» НГТУ г. Новочеркасска за предоставленную возможность использовать данные эксплуатации стальных канатов, а именно статистическую базу данных типовых повреждений стальных канатов.

Для цитирования. Хван Р.В. Сравнительный анализ качества работы искусственных нейронных сетей для оценки технического состояния стального каната. *Безопасность техногенных и природных систем*. 2024;8(2):68–77. https://doi.org/10.23947/2541-9129-2024-8-2-68-77

Introduction. Currently, methods for assessing the technical condition of engineering facilities using artificial neural networks (ANN) are becoming more widely used. This is due to the ease of identifying dependencies between the output data, which in this case is the technical condition of steel ropes, and the input data, which are various combinations of indicators that indicate defects in the steel ropes. To establish the correlation between the technical condition of a steel rope and various combinations of ten defective indicators expressed as a percentage of the permissible level of damage, a significant amount of analytical work and solving a multifactorial regression problem is required. Software that uses a neural network as its core can help experts and beginners in making decisions about the future operation of steel ropes based on different combinations of defect indicators.

Scientists from various fields of expertise are addressing the issues of reliability and safety in technical systems by employing modern artificial intelligence tools. Examples include the works of V.A. Vorontsov, E.A. Fedorov, A.A. Korotky, A.V. Panfilov, N.N. Nikolaev, A.R. Yusupov, S.V. Zhernakov, T.I. Goreva, N.N. Portyagin, G.A. Pyukke, B.Ch. Meskhi, A.N. Beskopylny, S.A. Stelmakh, I.F. Razveeva et al. [1–8]. These researchers successfully employ neural network modeling techniques to achieve various research and industrial objectives, such as evaluating the technical condition of aircraft engines and spacecraft systems, as well as detecting and classifying defects in steel ropes. At the same time, various architectures, ANN training parameters, software packages and artificial neural network development environments are used, such as Alyuda NeuroIntelligence, STATISTICA, Amygdala, and MatLab Simulink. It is also common practice to write your own program code using free (open-source) interactive libraries, such as TensorFlow, PyTorch, and Sikit-learn. However, it is worth noting that none of these works compare the quality of ANNs work, which have the same architecture, training parameters, volume and sample size, but are built using different methods, i.e. in different software development environments.

Often, the use of artificial neural networks on an industrial scale is associated with the use of expensive specialized software systems. These systems have an environment adapted for the implementation of ANNs and do not require users to have programming skills. A more affordable way to build and train a neural network from a financial point of view is to write your own software code using free interactive open source libraries. However, this method requires the user to have basic programming skills. The Scikit-learn library is open source, while the annual license for the STATISTICA software program costs about $25,000.

The aim of this work was to conduct a comparative analysis of the ANN quality in order to assess the technical condition of steel ropes by a combination of defective indicators built and trained by various methods, according to the criteria of test network performance, labor intensity and speed of network training, the average activation level of the "winning" neurons of the network and taking into account the financial costs of implementing and using the built artificial neural networks. In this regard, the task was set to create a new software tool for assessing the technical condition of steel ropes based on a combination of rejection criteria using two different types of ANNs: those built on the STATISTICA software package and those developed using the Scikit-learn interactive library.

Materials and Methods. The basis for ANNs training was the experience gained from operating steel ropes. This included a statistical database of typical damage to steel ropes and an expert assessment of their technical condition [9–12]. Figure 1 shows a neural network diagram that was used to assess the technical condition of a steel rope in all the methods described below.

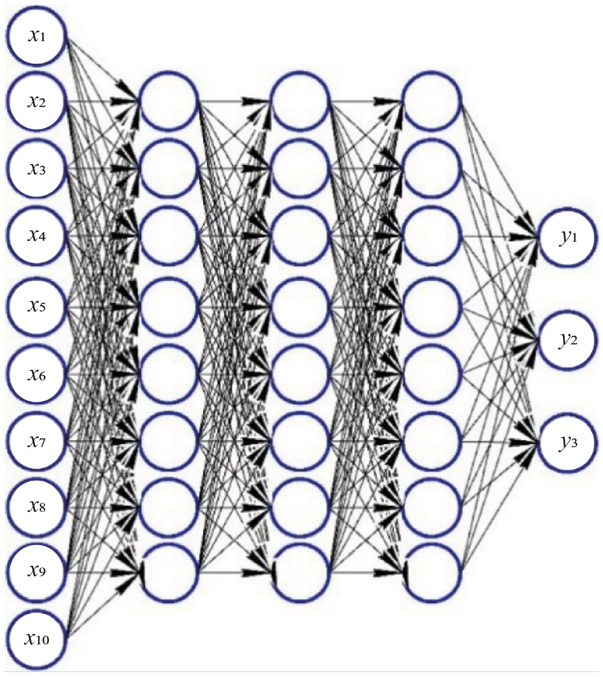


Fig. 1. ANN scheme for assessing the technical condition of a steel rope

Below are the parameters used in both methods of building the ANNs:

1. The so-called hyperparameters were set — parameters that did not change during network training. They included:

– 10 neurons of the input layer, equal to the number of defective parameters of the steel rope (x_1–x_{10});

– the number of output parameters (neurons), three possible states of the steel rope: 1 — operable, operation was allowed (y_1), 2 — defects within acceptable limits, operation was allowed with restrictions (y_2), 3 — the limit state had been reached, operation was prohibited (y_3);

– the number of intermediate layers, as well as the number of neurons in each intermediate layer.

Thus, the architecture of the neural network was defined [13].

2. The trainable parameters were set — parameters that were changed (optimized) in the process of network training: the values of synaptic weights *w* (the strength of the connection between neurons) and biases *b*. First, these values were set randomly, and then in the process of training the neural network, they were optimally configured.

3. The forward propagation algorithm was implemented by calculating a neural network based on randomly generated parameters *w* and *b*. To do this, training data was required, which consisted of samples of input parameters along with their corresponding known output parameters. These samples are shown in Table 1.

Table 1

Data for neural network training

No.	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
1	75	0	18	0	0	0	44	0	0	0	3
2	21	0	34	0	0	18	0	0	0	0	1
3	0	89	0	66	0	0	0	0	74	0	3
4	78	0	12	0	0	0	0	0	0	0	2
5	55	90	0	0	0	10	0	0	7	0	3
6	0	43	24	0	0	0	0	0	0	34	1
7	0	65	90	0	57	13	0	100	81	0	3
8	19	0	0	0	13	3	0	0	0	0	1
9	0	56	0	30	0	0	0	0	0	0	1
10	0	61	0	0	0	0	0	83	0	0	2
……	0	0	71	0	87	0	0	41	0	0	2
300	0	0	28	14	0	0	28	100	0	0	3

In Table 1, ten input parameters (x_1–x_{10}) represented the rejection values for steel ropes, expressed as a percentage of their limit values. This took into account a combination of various defects, which could be identified by expert means using a 3d model of the stress-strain state of the rope. Each combination corresponded to a specific technical condition of the rope. The volume of the training sample was 300 samples, while the control sample consisted of 30 samples. Ten test samples were also prepared for testing the ANN, but these were not included in the training process.

4. The activation function F was defined. This function was necessary to introduce non-linearity and a certain threshold value at the output of each neuron. We used the Relu activation function for this purpose. The formula below determined the level of neuron activation [14, 15]:

$$y = F\left(x_1 w_1 + x_2 w_2 + x_3 w_3 + b\right),$$

where y — level of neuron activation; x_1–x_3 — levels of activation of neurons of the previous layer; w_1–w_3 — synoptic weights; b — function offset.

Figure 2 shows a diagram of the formation of the activation level of one neuron.

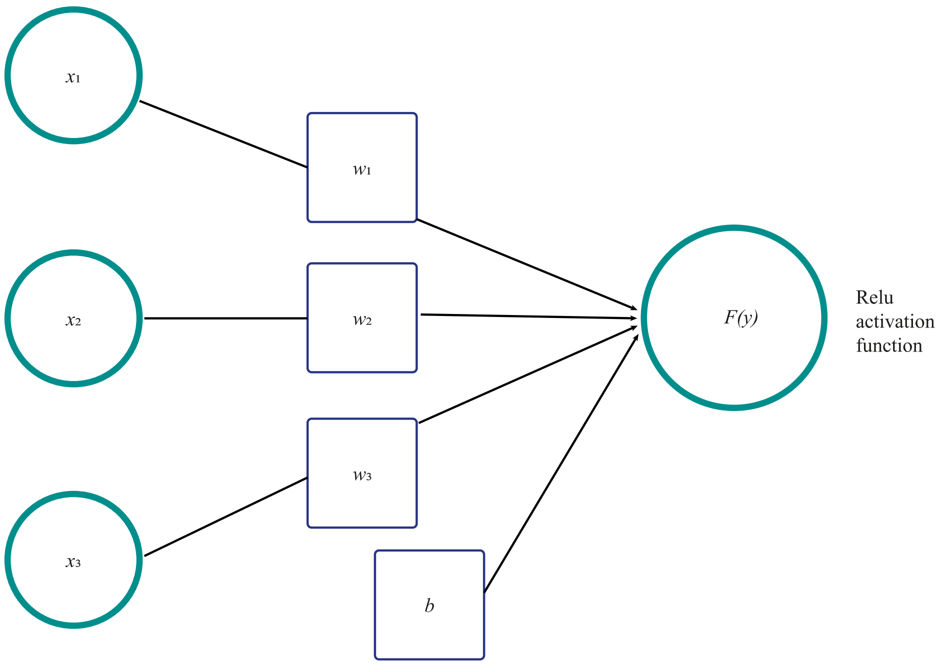


Fig. 2. Scheme of formation of the activation level of one neuron

5. A normalizing transformation of the activation values of the neurons in the last layer of the resulting network was performed using the softmax function, so that their values ranged from 0 to 1 and the sum of these values equaled 1. This transformation allowed for the interpretation of the activation levels of the neurons in a probabilistic sense. As a result, a prediction was made by the neural network — the state of the steel rope with a certain level of confidence for the current values of the synaptic weights (connections between neurons), *w*, and biases, *b*.

6. Calculation of error E between the calculated activation levels of neurons of output layer $y_{выч}$ and the target activation values of neurons of output layer $y_{цел}$ using the *MSE* (Euclidean distance) function or cross-entropy functions (used to determine the distance between probability distributions). It was important to remember that a neural network was trained with a teacher, which meant it used examples with known inputs and outputs. These examples were called training samples.

7. Implementation of the backpropagation algorithm, the purpose of which was to solve the problem of minimizing error function E depending on synaptic weights (connections between neurons) w and biases b. The gradient descent method was used for this purpose.

8. Repeating the entire learning algorithm on the next training sample (or group of samples) in order to minimize the error of the neural network by updating the connections (weights) between neurons. Each such repetition was called an epoch of learning. We set 2000 epochs of learning. The learning rate was 0.0001. The Adam error function. After completing a set number of training epochs, the neural network with all settings was saved and could be used to predict output parameters (the state of the steel rope) based on input parameters (combinations of defective indicators) that were not previously used in the network training process.

STATISTICA contains two built-in programming languages: STATISTICA BASIC and SCL (command language). The method of building a neural network in STATISTICA began with the launch of the "Neural Networks" module in the "Data Mining" tab on the main working panel of the program. The interface of the STATISTICA program practically did not differ from the well-known interface of the MS Office program. To build an ANN in STATISTICA, we set the network architecture and learning parameters specified in the above algorithm by pressing the appropriate keys using the adapted program interface.

The direct training of the neural network was activated by pressing the "Train" key in the STATISTICA working window, after which the ANN was checked on pre-determined control samples. The size of the control sample was 30. The condition for stopping the training process was when a benchmark network performance of at least 95% was achieved. The learning process of the neural network took about 20 seconds.

A step-by-step algorithm for writing the program code of a neural network using the interactive library Scikit-learn in the Python programming language was provided [4–7]:

1. In the first step, additional interactive libraries were installed. A library in this context meant a set of pre-written routines that made the programming process easier. To install a library, you need to enter the following code: (The explanations of the code or the commands being executed are provided in parentheses):

!python -m pip install pandas (data processing and analysis library);

!python -m pip install sklearn (machine learning library);

!python -m pip install openpyxl (library for working with Excel files).

2. In the second block, modules were imported by entering the following code:

import pandas as pd (data processing and analysis module);

from sklearn.neural_network import MLPClassifier (module for working with neural networks);

from sklearn.metrics import confusion_matrix, classification_report (additional module for working with neural networks);

import pickle (module used to save a trained neural network);

import joblib (module used to save a trained neural network).

3. Loading a database, i.e. a set of training data. It was necessary to prepare an Excel file with training data in advance, arranged in the format shown in Table 1 (a more simplified view is recommended — only columns of input and output parameters with their headers). To do this, we entered the following program code:

ds = pd.read_excel('book1.xlsx'), where book1.xlsx — an Excel file that should be located in the same folder as the file of the program being created;

ds.head(10) (code for visually displaying the first 10 rows of the table).

4. Next, values were assigned to the variables X and Y by writing the following program code:

X = ds.drop('Rope condition',axis=1) (all columns were assigned to variable X, except for column "Rope condition" (the literal name of the column was indicated);

y = ds['Rope condition'] (for variable Y, column "Rope condition" was assigned).

5. Building and training a neural network by writing the following code (this two-line program code replaced writing the neural network program code "manually" from the previous algorithm):

nn=MLPClassifier(hidden_layer_sizes=(8,8,8), max_iter=2000) (the architecture of the neural network was set, as well as the parameters of its training; in this case, 3 hidden layers, each with 8 neurons; the number of training epochs was 2000; by default, the Relu activation function was selected (code: activation='relu'), Adam error function (code: solver='adam'), learning rate 0.0001 (code: alpha=0.0001). It was possible to select various parameters of neural network training by writing the appropriate program codes separated by commas. All kinds of activation functions, error functions and other network parameters, their codes can be found on the official website of the interactive library Scikit-learn;

nn.fit(X, y) (introduction into the neural network of previously set parameters X — input variables (rejection indicators of steel ropes) and Y — output target variable (technical condition of steel ropes).

After launching this block, the neural network was built according to a given architecture and trained according to the specified learning parameters on training data (samples) uploaded via an Excel file table. The condition for stopping the training process was when a benchmark network performance of 95% was achieved. The learning process took approximately 1.5 minutes.

The following were the steps to save and use the ANN using the Scikit-learn interactive library:

1. Saving the trained neural network to a separate file by writing the following program code:

joblib_file = "joblib_model.pkl" (creating a joblib_model.pkl file);

joblib.dump(nn, joblib_file) (saving the nn neural network in the joblib_model.pkl file).

After running this block, the program created a joblib_model.pkl file in the same folder where the program file itself was located.

2. Creating a new file in PyCharm with py permission or in Jupyter with ipynb permission (the file in Jupyter was created for convenience, then the file would have to be saved in py format)).

3. Importing (to a new file) interactive libraries by writing the following program code:

import joblib (library for saving and embedding individual fragments of the program code);

import PySimpleGUI as sg (library for graphic design of the program).

4. Uploading a trained neural network to a new file via the joblib_model.pkl file, writing the following program code:

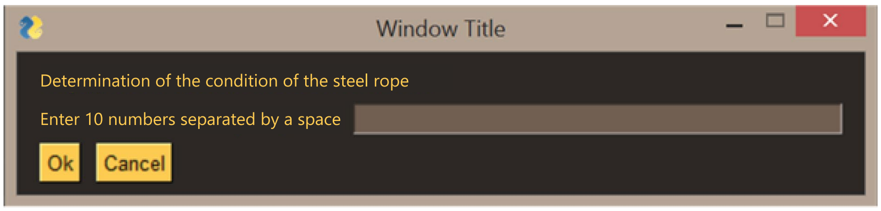
joblib_file = “D:\Python\PycharmProjects/joblib_model.pkl” (specifying the full path to the joblib_model.pkl file (individually for each user).

joblib_nn = joblib.load(joblib_file) (loading program code with a trained neural network).

5. Developing the graphical design of the program by writing the following program code:

```
sg.theme('DarkAmber')
layout = [ [sg.Text('Determination of the condition of the steel rope')];
[sg.Text('Enter 10 numbers separated by a space'), sg.InputText()];
[sg.Button('Ok'), sg.Button('Cancel')]
window = sg.Window('Window Title', layout)
while True:
event, values = window.read()
if event == sg.WIN_CLOSED or event == 'Cancel':
break
#Neural network
Xnew = [list(map(int, values[0].split()))]
y = joblib_nn.predict(Xnew)
#Popup window
if event == 'Ok':
window.disappear()
sg.popup('Rope condition ', y)
window.reappear()
print("Rope condition ', y)
window.close()
```

After starting this block, the program displayed the following working window (Fig. 3).



Window Title

Determination of the condition of the steel rope

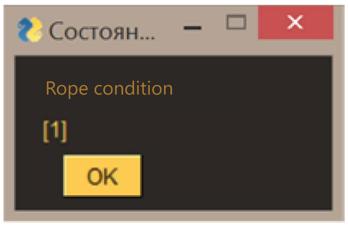
Enter 10 numbers separated by a space

Ok Cancel

Fig. 3. Working window of the written program

In the working window of the program there was a line for entering 10 numbers separated by a space. In this line, the actual percentages of ten different defects of steel ropes from their permissible values were entered. After entering, for example, the following values [20 0 30 0 0 10 0 0 0 0] and pressing the “Ok” key, the program took the user to the next working window.

In the working window (Fig. 4), the program displayed the information message “Rope condition [1]”, which corresponded to the rope condition “serviceable, operation is allowed”.



Состоян...

Rope condition

[1]

OK

Fig. 4. Working window of the program with a determined rope condition

Results. Thus, two ANNs have been developed that have the same architecture, training parameters, composition, number and volume of training, control and test samples, but built using different methods. The results of work of the ANN, built in the STATISTICA software package, are presented in Table 2.

Table 2

Results of work of the ANN built in STATISTICA

Sampling (test)	Target	Network output	Rope condition-1 (confidence level)	Rope condition-2 (confidence level)	Rope condition-3 (confidence level)
1	1	1	0.64	0.18	0.18
2	2	2	0.15	0.73	0.12
3	2	2	0.24	0.69	0.15
4	3	3	0.03	0.28	0.69
5	3	3	0.15	0.21	0.64
6	1	1	0.55	0.31	0.14
7	3	3	0.03	0.21	0.76
8	1	1	0.68	0.21	0.11
9	2	2	0.26	0.61	0.13
10	2	2	0.14	0.53	0.33

The column “Rope condition — Target” shows the previously known state of the rope, in the column “Rope condition — Output” — the result of the neural network; subsequent columns indicate the confidence probabilities of determining a particular condition of the rope, which correspond to the activation levels of the output neurons of the network.

According to Table 2, it can be seen that the neural network correctly determined the condition of the steel rope according to the defective indicators in 10 out of 10 cases, i.e. the test performance was 100%. At the same time, the average value of the confidence levels for determining the condition of the ropes was 0.65.

The results of work of the ANN, built using the interactive library Scikit-learn, are presented in Table 3.

Table 3

Results of work of the ANN built using Scikit-learn

Sampling (test)	Target	Network output	Rope condition-1 (confidence level)	Rope condition-2 (confidence level)	Rope condition-3 (confidence level))
1	1	1	0.58	0.13	0.29
2	2	2	0.19	0.51	0.19
3	2	2	0.22	0.56	0.30
4	3	3	0.29	0.25	0.46
5	3	3	0.21	0.22	0.57
6	1	1	0.57	0.22	0.21
7	3	3	0.21	0.22	0.57
8	1	1	0.63	0.22	0.15
9	2	2	0.22	0.57	0.21
10	2	2	0.33	0.45	0.21

According to Table 3, it can be seen that the neural network correctly determined the condition of the steel rope according to the defective indicators in 10 out of 10 cases, i.e. the test performance was 100%. At the same time, the average value of the confidence levels for determining the condition of the ropes was 0.55.

Comparisons of test performance of neural networks with the same architecture and learning parameters, but built using different methods, showed that the ANN built on the basis of the STATISTICA software package and the ANN built using the Scikit-learn interactive library had a test performance of 100% with a test sample size of 10. However, the average confidence level (activation level of the “winning” neuron) for determining the state of the steel rope of the ANN built on the basis of the STATISTICA software package was 0.65, whereas the same indicator of the ANN built using the interactive library Scikit-learn was 0.55, which is 15% less.

Discussion and Conclusion. The results obtained showed that the ANN built using the STATISTICA software package, with the same architecture and network learning parameters, had more optimal software algorithms according to the criteria of confidence in assessing the technical condition of the steel rope and the speed of network learning, in comparison with the ANN built using the free Scikit-learn library. This can be explained by the fact that when developing algorithms for software complexes such as STATISTICA, specialized hardware complexes are used, including vector and tensor processors, which go far beyond the capabilities of the average application developer and require the involvement of highly qualified specialists. However, the indicator of the test performance of the ANN turned out to be the same for both ANNs. It is important to note that when evaluating this indicator, the test sample size was 10. If the test sample is increased, this indicator will be more accurate. At the same time, the achieved result justifies the use of TensorFlow, PyTorch, and Scikit-learn libraries by the world's leading research and commercial centers in the field of artificial intelligence.

In addition, the obtained scientific conclusion will allow us to numerically evaluate and compare the quality of ANNs having the same architecture and learning parameters, but built using different methods, and may be useful both for future scientific research in this field and for choosing the optimal environment for building ANNs in the industrial sphere. The developed programs can be used by specialists and experts as intelligent decision support systems for diagnosing the technical condition of steel ropes.

References

1. Zhernakov SV. Application of Neural Network Technology to Diagnose the Technical Condition of Aircraft Engines. *Intellektual'nye Sistemy v Proizvodstve*. 2006;2(8):70–83. (In Russ.).

2. Panfilov AV, Nikolaev NN, Khvan RV, Korotkiy AA. Assessment of Possible Cable Car Accidents by Employee Competencies Using Neural Networks. *Nauchno-Tekhnicheskiy Vestnik Bryanskogo Gosudarstvennogo Universiteta*. 2023;(1):79–86. https://doi.org/10.22281/2413-9920-2023-09-01-79-86 (In Russ.).

3. Goreva TI, Pornjagin NN, Pjukke GA. Neural Network Model Diagnosis Technical Systems. *Bulletin of the Kamchatka Regional Association Educational and Scientific Center (KRASEC). Physicsal and Mathematicsal Sciences.* 2012;1(4):31–43. (In Russ.).

4. Beskopylny AN, Shcherban EM, Stelmakh SA, Mailyan LR, Meskhi B, Razveeva I, et al. Discovery and Classification of Defects on Facing Brick Specimens Using a Convolutional Neural Network. *Applied Sciences.* 2023;13(9):5413. https://doi.org/10.3390/app13095413

5. Stelmakh SA, Shcherban EM, Beskopylny AN, Mailyan LR, Meskhi B, Razveeva I, et al. Prediction of Mechanical Properties of Highly Functional Lightweight Fiber-Reinforced Concrete Based on Deep Neural Network and Ensemble Regression Trees Methods. *Materials.* 2022;15(19):6740. https://doi.org/10.3390/ma15196740

6. Beskopylny AN, Stelmakh SA, Shcherban EM, Mailyan LR, Meskhi B, Razveeva I, et al. Concrete Strength Prediction Using Machine Learning Methods CatBoost, k-Nearest Neighbors, Support Vector Regression. *Applied Sciences.* 2022;12(21):10864. https://doi.org/10.3390/app122110864

7. Beskopylny AN, Shcherban EM, Stelmakh SA, Mailyan LR, Meskhi B, Razveeva I, et al. Detecting Cracks in Aerated Concrete Samples Using a Convolutional Neural Network. *Applied Sciences.* 2023;13(3):1904. https://doi.org/10.3390/app13031904

8. Vorontsov VA, Fedorov EA. Development of a Prototype of an Intelligent System for Operational Monitoring and Technical Condition of the Main Onboard Systems of the Spacecraft. *Trudy MAI.* 2015;2:1–35. (In Russ.).

9. Panfilov AV, Meskhi BCh, Korotkiy AA, Yusupov AR, Khvan RV. *Software and Hardware Complex for Visual and Measuring Control of Steel Ropes Based on Computer Vision and Artificial Intelligence.* Monograph. Rostov-on-Don: DSTU; 2023. 131 p. (In Russ.).

10. Panfilov AV, Nikolaev NN, Yusupov AR, Korotkiy AA. Integral Risk Assessment in Steel Ropes Diagnostics Using Computer Vision. *Safety of Technogenic and Natural Systems.* 2023;(1):56–69. https://doi.org/10.23947/2541-9129-2023-1-56-69

11. Seyed Reza Ghoreishi, Tanguy Messager, Cartraud P, Davies P. Validity and Limitations of Linear Analytical Models for Steel Wire Strands under Axial Loading, Using a 3D FE Model. *International Journal of Mechanical Sciences.* 2007;49(11):1251–1261. https://doi.org/10.1016/j.ijmecsci.2007.03.014

12. Frikha A, Cartraud P, Treyssède F. Mechanical Modeling of Helical Structures Accounting for Translational Invariance. Part 1: Static Behavior. *International Journal of Solids and Structures.* 2013;50(9):1373–1382. https://doi.org/10.1016/j.ijsolstr.2013.01.010

13. Korotkiy AA, Panfilov AV, Khvan RV, Yusupov AR. Integral Method of Assessing Defects on the Operability of Steel Rope Using Artificial Neural Networks. *Transport, mining and construction engineering: science and production.* 2023;8:73–79. https://doi.org/10.26160/2658-3305-2023-18-73-79 (In Russ.).

14. Foti F, De Luca di Roseto A. Analytical and Finite Element Modelling of the Elastic–Plastic Behaviour of Metallic Strands under Axial–Torsional Loads. *International Journal of Mechanical Sciences.* 2016;115–116:202–214. https://doi.org/10.1016/j. ijmecsci.2016.06.016

15. Spak K, Agnes G, Inman D. Cable Modelling and Internal Damping Developments. *Applied Mechanics Reviews.* 2013;65(1):010801. https://doi.org/10.1115/1.4023489

About the Author:

Roman V. Khvan, Cand.Sci. (Eng.), Senior Lecturer of the Operation of Transport Systems and Logistics Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, RF), SPIN-code: 8662-6094, ORCID, ResearcherID, ScopusID, khvanroman@yandex.ru

Conflict of Interest Statement: the author does not have any conflict of interest.

The author has read and approved the final version of the manuscript.

Received 12.03.2024
Revised 05.04.2024
Accepted 11.04.2024

Об авторе:

Роман Владимирович Хван, кандидат технических наук, старший преподаватель кафедры эксплуатации транспортных систем и логистики Донского государственного технического университета (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), SPIN-код: 8662-6094, ORCID, ResearcherID, ScopusID, khvanroman@yandex.ru

Конфликт интересов: автор заявляет об отсутствии конфликта интересов.

Автор прочитал и одобрил окончательный вариант рукописи.

Поступила в редакцию 12.03.2024
Поступила после рецензирования 05.04.2024
Принята к публикации 11.04.2024