Technical University of Košice

Faculty of Mining, Ecology, Process Control and Geotechnologies

Student Scientific Conference

Section S2A Control and Logistics

Automatický tvorca predikčných modelov

Automatic prediction model builder

Name::	Bc. Aleš Jandera
Level of study:	II.
Year of study:	2.
Study Programme:	Process control of raw materials and material extraction and processing
Supervisor:	doc. Ing. Tomáš Škovránek, PhD.

Abstract in English

Neural network data classification has been successfully applied to a wide range of data classification problems. This paper presents a Neural Network Data Classification model builder system in the form of an application with a user-friendly user interface. The application is based on a combination of machine learning algorithms with a statistical approaches such as Monte Carlo and Markov chains. The classification of the data is carried out using a machine learning algorithm, which provides the optimal solution for the creation of the prediction model. Application is created in Matlab.

Keywords in English

Mathematic modeling, forecasting, linear prediction, neural netowrk, prediction model builder

Abstract in Slovak

Klasifikácia údajov neurónovej siete bola úspešne aplikovaná na širokú škálu problémov klasifikácie údajov. Táto práca predstavuje systém na tvorbu predikčých modelov vo forme aplikácie s užívateľsky prívetivým užívateľským rozhraním. Aplikácia je založená na kombinácii algoritmov strojového učenia so statistickými metódami ako sú Monte Carlo a Markovove reťazce. Klasifikácia údajov sa vykonáva pomocou algoritmu strojového učenia, ktorý poskytuje optimálne riešenie problému klasifikácie v aplikácii Matlab.

Keywords in Slovak

Matematické modelovanie, predpoved, lineárna predikcia, neurónová sieť

Contents

List of Figures

1 Analysis

1.1 Mathematical models for data prediction

Mathematical prediction models are tools used to forecast the behavior of a system or process. They are typically built using mathematical equations or algorithms that are designed to describe the relationship between the input and output variables of the system.

There are several types of mathematical prediction models, including linear regression, time series analysis, and machine learning algorithms such as neural networks and decision trees. Each type of model has its strengths and weaknesses, and the choice of model depends on the specific problem being addressed.

Linear regression models are used to describe the relationship between two or more variables by fitting a straight line to the data. Time series analysis is used to predict future values of a variable based on its past values, and can be used to forecast trends, seasonal patterns, and other patterns in time series data.

Machine learning algorithms are increasingly being used for prediction modeling, as they can learn complex relationships between input and output variables and adapt to changing data patterns. Neural networks, for example, are designed to simulate the structure and function of the human brain and can be used for tasks such as image recognition, natural language processing, and predicting the outcome of events.

Mathematical prediction models are used in a wide range of fields, including finance, economics, engineering, and the natural sciences. They can be used to forecast stock prices, predict the spread of disease, optimize industrial processes, and much more.

1.1.1 Regresion models

Regression models are a type of statistical models used to examine the relationship between a dependent variable and one or more independent variables [1]. The goal of regression analysis is to model the relationship between these variables and make predictions about the dependent variable based on the values of the independent variables. Regression models are widely used in many fields, including economics, finance, marketing, and social sciences, to make predictions and understand the relationship between variables. There are several types of regression models, including:

- Linear regression is a simple regression model where the relationship between the dependent and independent variables is modeled using a linear equation.
- Logistic regression is used for binary classification problems where the dependent variable is binary and the goal is to model the relationship between the independent variables and the probability of the dependent variable being either 0 or 1.
- Multiple regression is used when there are multiple independent variables and the goal is to model the relationship between all of these variables and the dependent variable.
- Polynomial regression is used when the relationship between the dependent and independent variables is non-linear and can be modeled using a polynomial equation.

The choice of regression model depends on the nature of the data and the research question being asked.

1.1.2 Time-series models

Time-series models are mathematical models used to analyze and forecast data that are collected over time [2]. These models are used to study and make predictions about the trends, patterns, and behavior of the data over time, taking into account historical values and their relationship with the present. Time-series models are widely used in areas such as economics, finance, and weather forecasting, among others. The models are based on various statistical techniques, including ARIMA (AutoRegressive Integrated Moving Average), SARIMA (Seasonal ARIMA), and exponential smoothing, among others. The goal of time-series modeling is to build a mathematical representation of the underlying process that generates the time-series data, allowing for accurate prediction of future values. Time-series models are statistical models used to analyze and make predictions about time-dependent data. They are widely used in various fields, including finance, economics, engineering, and social sciences.

Time-series models make use of past values of a variable to predict future values. They assume that there is a pattern or trend in the data that can be used to forecast future behavior. Some commonly used time-series models include:

- Autoregressive Integrated Moving Average (ARIMA). This model is used to analyze and forecast stationary time-series data. It consists of three components: autoregression, differencing, and moving average.
- Seasonal Autoregressive Integrated Moving Average (SARIMA) is an extension of ARIMA that takes into account seasonal patterns in the data.
- Exponential Smoothing (ETS) is used to forecast time-series data that has a trend and/or seasonality. It uses a smoothing parameter to assign more or less weight to past observations based on their recency.
- Vector Autoregression (VAR) is used when there are multiple time-series variables that influence each other. It can be used to analyze the relationships between these variables and to make predictions about their future be havior.
- These models are valuable tools for analyzing and predicting time-series data, but they require careful consideration of the specific characteristics of the data being analyzed and the appropriate model to use.

1.2 Neural networks

A neural network is a type of machine learning algorithm inspired by the structure and function of biological neurons in the human brain. It is composed of interconnected nodes, called neurons, that are organized into layers. The input layer receives raw data, such as images or text, and passes it on to the hidden layers, which perform calculations and apply weights to the input data to create a prediction. Finally, the output layer produces the final prediction or classification.

As we can see on image 1.1 each input X_n should be properly weighted by a certain weight W_n before all the signals enter the summation stage. Afterwards, the weighted summation is forwarded into the activation unit producing the neuron's output signal.

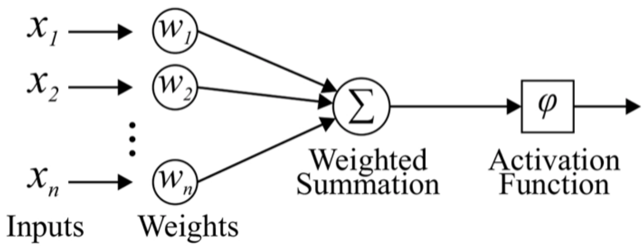


Figure 1.1: Perceptron preview. [3]

Neural networks are trained on large datasets using a process called backpropagation, which adjusts the weights and biases of the neurons to minimize the error between the predicted output and the actual output. Once a neural network has been trained, it can be used to make predictions on new data.

A neuron is a basic building block of a neural network, also known as an artificial neuron or a perceptron. It is modeled after the biological neuron in the human brain, which receives input signals from other neurons, processes them, and sends output signals to other neurons.

In a neural network, a neuron receives input from other neurons or directly from the input data, applies a mathematical function to the input, and produces an output that is sent to other neurons in the network. The input to a neuron is usually a vector of numbers, and each input is multiplied by a corresponding weight. The neuron then sums up the weighted inputs, adds a bias term, and applies an activation function to the result.

The purpose of the activation function is to introduce nonlinearity into the neuron, which allows the neural network to learn complex patterns and relationships in the data. There are several different types of activation functions that can be used, such as the sigmoid function, ReLU (Rectified Linear Unit) function, and tanh (hyperbolic tangent) function.

The output of a neuron is typically fed into other neurons in the next layer of the neural network. The weights and biases of the neurons are adjusted during the training process using a technique called backpropagation, which involves computing the gradient of the error with respect to the weights and updating them using an optimization algorithm such as stochastic gradient descent.

Feedforward Neural Networks
These are the most basic type of neural networks, where the information flows only in one direction,from input to output. These networks can have one or more hidden layers and are often used for classification or regression tasks. Shema on basic feedforward NN is on fig 1.2

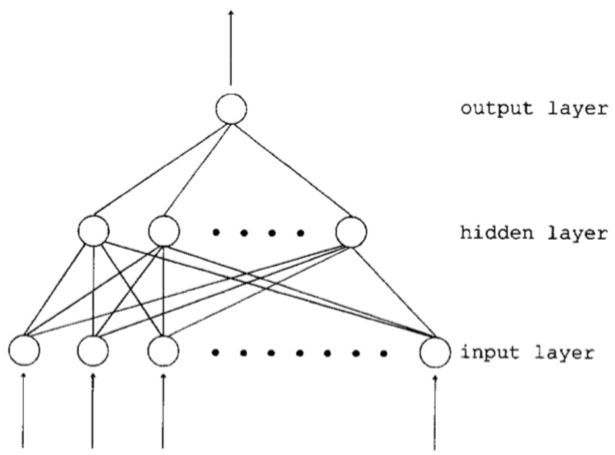
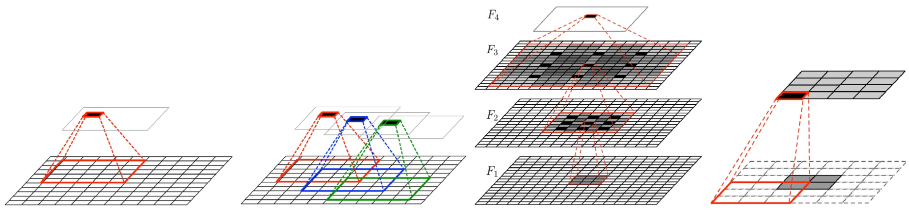


Figure 1.2: Typical feed-forward neural network composed of three layers. [4]

Convolutional Neural Networks (CNNs)
These networks are specialized for processing images and are commonly used in computer vision tasks. They use convolutional layers to extract features from images and can learn to recognize patterns and objects in images see in fig 1.3

Recurrent Neural Networks (RNNs)
These networks are designed to work with sequential data, such as time-series



(a) Convolution (b) Tiled Convolution (c) Dilated Convolution (d) Deconvolution

Figure 1.3: Illustration of (a) Convolution, (b) Tiled Convolution, (c) Dilated Convolution, and (d) Deconvolution. [5]

or natural language data. They have loops that allow information to be passed from one time-step to the next, enabling them to capture temporal dependencies in the data, described on fig 1.4

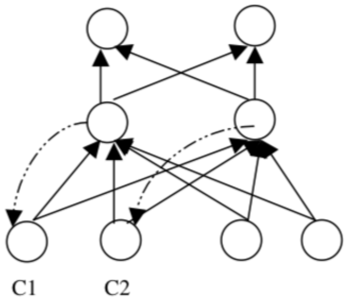


Figure 1.4: Typical recurrent network. [6]

Long Short-Term Memory Networks (LSTMs)
These are a type of RNN that are designed to address the problem of vanishing gradients in traditional RNNs. They use memory cells and gates to selectively retain or forget information over time, making them well-suited for learning from long sequences. As you can see on fig 1.5 color indicates degree of memory activation.

Autoencoder Neural Networks
These networks are used for unsupervised learning and are designed to learn a compressed representation of the input data. As we can see on figure 1.6 they consist of an encoder that maps the input data to a compressed representation, and a decoder that maps the compressed representation back to the original data.

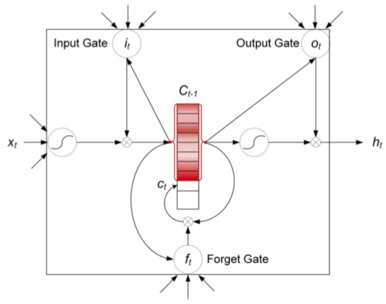


Figure 1.5: Long short-term memory network. [7]

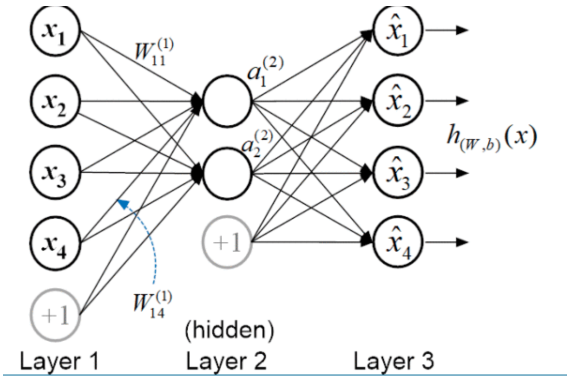


Figure 1.6: An autoencoder neural network. [8]

Generative Adversarial Networks (GANs)

These networks consist of two networks, a generator and a discriminator (see on figure 1.7), that are trained together in a game-theoretic framework. The generator is trained to generate realistic data samples, while the discriminator is trained to distinguish between real and generated data samples.

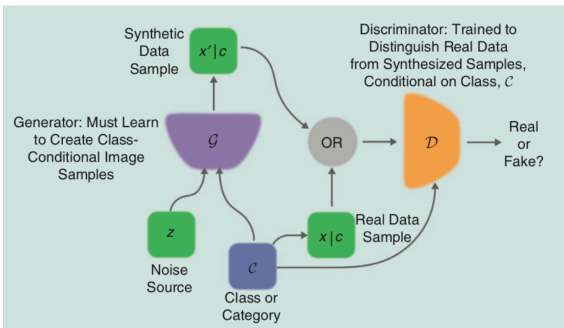


Figure 1.7: The conditional GAN schema. [9]

1.2.1 Classification

Neural network data classification is a technique for categorizing data into different classes or categories based on patterns and features present in the data. a neural network is a type of machine learning algorithm that is modeled after the structure and function of the human brain. It is composed of interconnected nodes or neurons that are organized into layers.

In a classification task, the neural network is trained on a dataset that is labeled with the correct class for each example. During training, the network learns to recognize patterns and features in the input data that are associated with each class. The process of training involves adjusting the weights and biases of the neurons in the network to minimize the error between the predicted class and the actual class of each example in the training set.

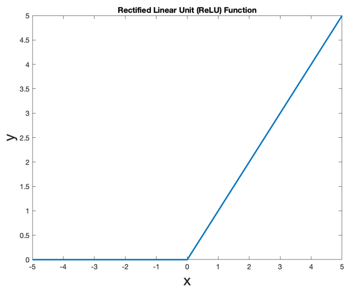
Neural network data classification has been successfully applied to a wide range of tasks, including image classification, speech recognition, natural language processing, and fraud detection, among others [10].

1.2.2 Activation functions

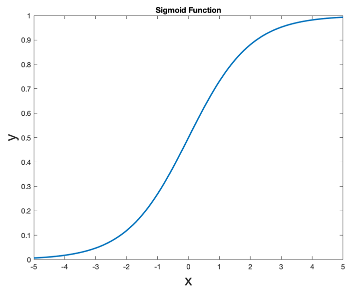
There are several types of activation functions [11] used in neural networks, as we can see on figure 1.8 including:

- Sigmoid Function: the sigmoid function is a commonly used activation function that maps any input value to a value between 0 and 1. It is typically used in binary classification problems and in the output layer of neural networks that produce probability estimates 1.8b.
- ReLU (Rectified Linear Unit): the ReLU function is another popular activation function that maps any input value less than 0 to 0, and any input value greater than or equal to 0 to the input value itself. It is computationally efficient and has been shown to work well in deep neural networks 1.8a.
- Tanh Function: the tanh (hyperbolic tangent) function is similar to the sigmoid function, but it maps input values to a range between -1 and 1. It is commonly used in the hidden layers of neural networks 1.8c.
- Softmax Function: the softmax function is often used in the output layer of neural networks that produce multi-class classification predictions. It maps the outputs to a probability distribution over the possible classes 1.8d.

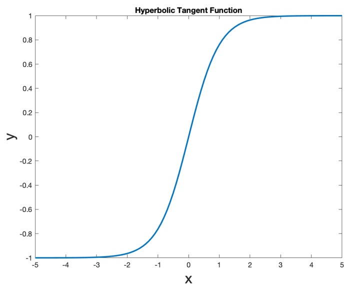
- Leaky ReLU: the Leaky ReLU function is similar to the ReLU function, but it allows a small, non-zero gradient when the input value is negative. This can help to prevent the "dying ReLU" problem, where some ReLU units become inactive and stop contributing to the network's output 1.8e.
- ELU (Exponential Linear Unit): the ELU function is similar to the ReLU function, but it allows negative values to have non-zero outputs. This can help to prevent the "dying ReLU" problem and can improve the performance of deep neural networks 1.8f.



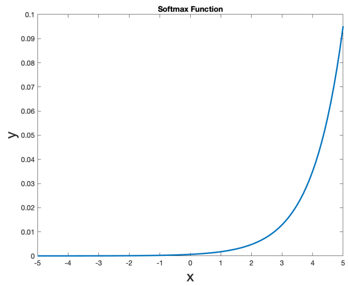
(a) ReLU



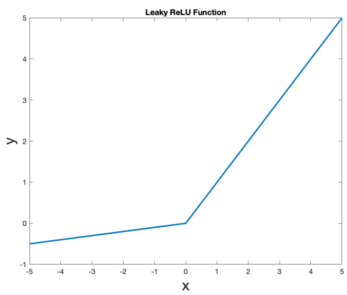
(b) Sigmoid



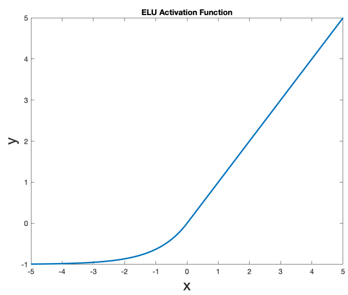
(c) Tanh



(d) Softmax



(e) Leaky ReLU



(f) ELU

Figure 1.8: Neural network activation functions.

These are some of the most commonly used activation functions in neural networks, but there are many other types of activation functions that have been developed for specific tasks or to address certain problems.

2 Implementation

Application which will all principles describes in 1. Created project is focused on the design and development of the Automatic prediction model builder system in the form of an application with a user-friendly user interface, which will be implemented in the cloud environment. As a development environment Matlab ecosystem was used.

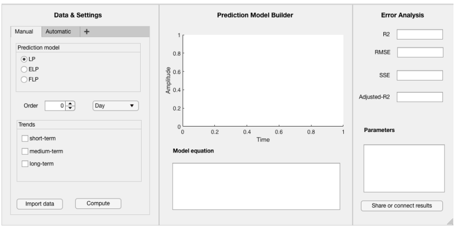
2.1 Mathematical models

Prediction models were based mainly on the principle of linear prediction (LP) and its modifications, such as non-integer linear prediction (fractional linear prediction - FLP), LP extended by parameters capable of capturing short-term and long-term trendiness in data (extended linear prediction - ELP), etc., which will be extended by further statistical methods such as Monte Carlo, Markov chains, etc.

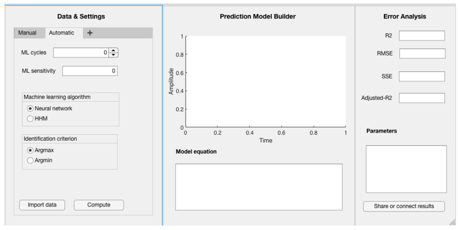
For the identification of the appropriate structure of economic and behavioral models and the identification of the parameters of the selected models, machine learning algorithms will be used, which will provide the optimal solution for the selected data and thus the use-case.

2.2 Application

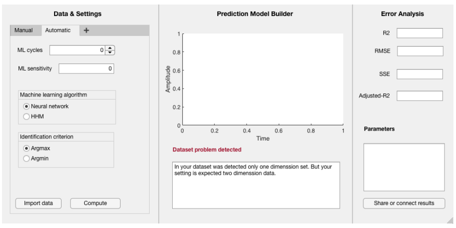
This application would make it possible to easily and accurately predict various socioeconomic macro and micro indicators, such as gross/net domestic/national product, economic wealth, unemployment, inflation, average/minimum wage, purchasing power of the population but also the behavior of customers (customers can also be perceived as households), intended for sectors such as public or state administration, public planning (but also private) finance, banking.



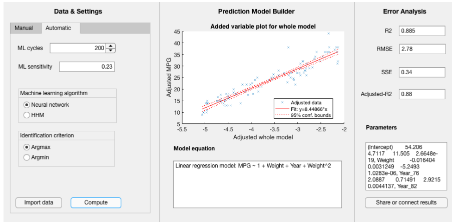
(a) Manual settings



(b) Automatic settings



(c) Dataset warning



(d) Results

Figure 2.1: Application overview

From the point of view of commercial use, a possible application would be predicting the number of customers and the number of orders, the company's income, the success of marketing strategies, or on the basis of the prediction, the planning of warehouse stocks.

2.2.1 Import datasets

First step to get results from predictive application. Use the button in the left bottom corner 2.1a to import our dataset. Application is able to process datasets in csv or excel format. Application is trying to identify your dataset by the setting 2.2.2. If the problem in our imported dataset is detected, application show warning 2.1c with the tips how to preparte your dataset in a better way.

2.2.2 Setting

Application has two way of settings, we can define parameters of our model manualy or we can choose automatic setting of application to detect the best prediction model based on inserted dataset.

Manual setting

Manual setting is to create a prediction model based on linear prediction. We are able to use these parameters:

- **Prediction model**
  Choose one of type from Linear prediction 2.2, Extended linear prediction 2.1,

Fraction linear prediction 2.3.

$$\hat{x}(n) = \left(\sum_{i=1}^{p} a_i x(n-i) + \sum_{i=1}^{q} b_i x(n-S-i) \right) * \gamma(n), \tag{2.1}$$

where $\hat{x}(n)$ is the predicted value of the order at time n, $x(n-i)$ are the past short-therm predition part p samples of the dataset, $x(n-S-i)$ are the past long-term prediction part with seasonal shift S, and a_i and b_i are the predictors coefficients. The order of the predictor is p for short-term and q for the long-term linear prediction. The seasonal weights is represent by $\gamma(n)$.

$$\hat{x}(n) = \sum_{i=1}^{p} a_i x(n-i), \tag{2.2}$$

where $\hat{x}(n)$ is the predicted value of $x(n)$, p is the order of the predictor, and a_i are the predictor coefficients. The predictor coefficients can be found by minimizing the prediction error.

$$\hat{x}(n) = a_1 x(n-i) + a_2 \frac{h^\alpha}{T(1+\alpha)} D^\alpha x(n-1), \tag{2.3}$$

that uses two-samples memory, is independent of the order of fractional derivative α [12].

- **Order of predictor**
  In this step we are able to set up the order of linear prediction and the period of predicted values.

- **Trends detection**
  In this setting you are able to choose the length of the period to identify the prediction model parameters and trends.

Automatic setting
Autoimation setting we can used for set up the parameters of neural network, which will be used to set up all optimal parameters for prediction. For set up automation part of application we are able to use these parameters:

- **Machine learning cycles**
  Number of observations use to find optimal parameters of the model.

- **Machine learning sensitivity**
  Sensitivity is a measure of how well a machine learning model can detect positive instances. It is also known as the true positive rate (TPR) or recall.

- **Machine learning alghoritm**
  In this section we are able to choose the neural network or hidden markov model to run in backround. HMM is statistical approach and in specific dataset can provide beter results.

- **Identification criteria**
  In this part we can choose the maximalization or minimalization as a function to found optimal parameters.

2.2.3 Results

Fo our test purpose we used the automatic setting of the application and try to run over our dataset. On the figure 2.1d we can see the results of our application. Application succefully detect the model equation, identify optinal parameters and calculate the comparison criteria. Model builder create the model with $R^2 = 0.885$. Our application is show the results in these tree sections:

Plot of results

On the figure 2.1d we can see the main window at the middle of the screen where the model results and loaded dataset are plotted.

Model equation

Under the main section with plotted results is the model equation section where created model by the application is shown.

Parameters

In the right bottom corner we see the parameters for our model. When we combine Model equation and Parameters section we are able to use the created model in other opliations and approaches as we need.

Error Analysis

The last part of the application is Error Analysis in where we can find the result of our created model. Application privides R^2, adjusted R^2, Root Mean Square Error and Sum Squared Error.

Bibliography

1. FAHRMEIR, Ludwig; KNEIB, Thomas; LANG, Stefan; MARX, Brian D. Regression models. In: *Regression: Models, methods and applications*. Springer, 2022, pp. 23–84.
2. CRYER, Jonathan D. *Time series analysis*. Duxbury Press Boston, 1986.
3. MOURGIAS-ALEXANDRIS, G.; TSAKYRIDIS, A.; PASSALIS, N.; TEFAS, A.; VYRSOKINOS, K.; PLEROS, N. An all-optical neuron with sigmoid activation function. *Opt. Express*. 2019, vol. 27, no. 7, pp. 9620–9630. Available from DOI: `10.1364/OE.27.009620`.
4. SVOZIL, Karl. Quantum logic. 1997.
5. GU, Jiuxiang; WANG, Zhenhua; KUEN, Jason; MA, Lianyang; SHAHROUDY, Amir; SHUAI, Bing; LIU, Ting; WANG, Xingxing; WANG, Gang; CAI, Jianfei; CHEN, Tsuhan. Recent advances in convolutional neural networks. *Pattern Recognition*. 2018, vol. 77, pp. 354–377. ISSN 0031-3203. Available from DOI: `https://doi.org/10.1016/j.patcog.2017.10.013`.
6. MEDSKER, Larry R; JAIN, LC. Recurrent neural networks. *Design and Applications*. 2001, vol. 5, pp. 64–67.
7. CHENG, Jianpeng; DONG, Li; LAPATA, Mirella. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*. 2016.
8. LUO, Tie; NAGARAJAN, Sai G. Distributed anomaly detection using autoencoder neural networks in WSN for IoT. In: *2018 ieee international conference on communications (icc)*. 2018, pp. 1–6.
9. CRESWELL, Antonia; WHITE, Tom; DUMOULIN, Vincent; ARULKUMARAN, Kai; SENGUPTA, Biswa; BHARATH, Anil A. Generative adversarial networks: An overview. *IEEE signal processing magazine*. 2018, vol. 35, no. 1, pp. 53–65.
10. FÉRAUD, Raphael; CLÉROT, Fabrice. A methodology to explain neural network classification. *Neural networks*. 2002, vol. 15, no. 2, pp. 237–246.

11. GÉRON, Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

12. DESPOTOVIC, Vladimir; SKOVRANEK, Tomas; PERIC, Zoran. One-parameter fractional linear prediction. *Computers & Electrical Engineering*. 2018, vol. 69, pp. 158–170. ISSN 0045-7906. Available from DOI: `https://doi.org/10.1016/j.compeleceng.2018.05.020`.