Neural networks and their application in forecasting problems

V A Ivanyuk¹² and F F Pashchenko³

¹ Department of Data Analysis and Machine Learning, Financial University under the Government of the Russian Federation, 49 Leningradsky Prospekt, 125993, Moscow, Russia
² Department of Higher Mathematics, Bauman Moscow State Technical University, Moscow, Russia
³ V.A. Trapeznikov Institute of Control Sciences of RAS, Moscow, Russia

E-mail: VAIvanyuk@fa.ru

Abstract. The report describes popular machine learning methods and applications of neural networks. It reveals methods of training neural networks and offers a method of forecasting based on neural networks for modelling financial time series. Neural networks have recently gained in popularity among scholars. In particular, neural networks are widely used in the field of visualization and image recognition. But the practical significance of neural networks does not end there, they also find usage in such areas as forecasting, classification, clustering and modelling. This success can mainly be attributed to the remarkable property of models based on neural networks – they can «see» non-linear connections in contrast to many models, which for the most part have linear connections only. Currently, the use of neural networks is developing in the following directions: Stock market and macroeconomic forecasting (Neuro XL, OptimuStock, StocksNeural); Speech recognition and man-machine dialogue (Siri, Alexa, Cortana, Alice); Imitation of intellectual activity (weak AI in Siri, Alexa, Cortana, Alice); Improving low-quality and noisy information (DeepImagePrior). The advantages of using neural networks include: versatility, simplicity. Neural networks are able to model dependencies also in cases when there is a large number of variables.

1. Introduction

Artificial neural networks are popular machine learning methods that simulate the learning mechanism of biological organisms [1]. The human nervous system contains cells called neurons. In fact, any neural network is a collection of neurons and connections between them. A neuron is a function with multiple inputs and one output. Its task is to take all the values from its input, perform operations on them, and send the result to the output, using weights as intermediate parameters. This sequence of operations applied to the input data matrix is usually a set of additions and multiplications followed by nonlinear functions.

An artificial neural network computes the input data function by propagating the computed values from the input neurons to the output neurons and using the weights as intermediate parameters. Learning occurs by changing the weight connecting the neurons [2]. Just as external stimuli are necessary for learning in biological organisms, external stimuli in artificial neural networks are provided by the training data containing examples of input-output pairs of the function to be learned. For example, the training data may contain pixel representations of images (input data) and their labels as the output data.
These pairs of the training data are fed into the neural network using the input representations to get predictions about the output labels. The training data provides feedback on the correctness of the weights in the neural network, depending on how well the predicted output for a particular input matches the training output label in the training data. The errors made by the neural network when computing the function can be considered as a kind of unpleasant feedback in a biological organism, leading to an adjustment in the synaptic strength. Similarly, in a neural network, the weights between neurons are adjusted in response to prediction errors. The goal of changing the weights is to modify the computed function in order to make predictions more correct in subsequent iterations. Therefore, the weights are carefully changed in a mathematically sound way to reduce the error in the computation. By sequentially changing the weights between neurons over many inputs and outputs, the function computed by the neural network is improved over time to provide more accurate predictions. Eventually, if a neural network learns many different images, it will be able to correctly recognize them from an image that it has not seen before. This ability to accurately compute functions of unseen inputs by training on a finite set of input-output pairs is called model generalization. The main benefit of all machine learning models is their ability to generalize their learning from previously seen training data to unseen examples [3, 4].

2. Peculiarities of neural networks

Neural network can be considered as a computational graph of elementary units, in which great power is achieved by connecting them in certain ways. When a neural network is used in its most basic form, without putting multiple units together, the learning algorithms are often reduced to classic machine learning models. The real value of a neural model over classical methods is revealed when these elementary computational units are combined, and the weights of the elementary models are trained using their dependencies on each other. By combining several units, one can increase the power of the model to learn more complex data functions, compared to those inherent in classic machine learning models [5, 6, 7].

A typical neural network can be divided into input, hidden, and output layers. The data is first received by the input layer, where extensive features are detected. Then the hidden layers analyze and process the data. Based on previous computations, the data is ordered as it is propagated through each hidden layer of the network. The final result is mapped to the output layer.

The middle layers are considered hidden because, like human vision, they implicitly divide objects between the input and output layers.

There are many methods of connecting the nodes of a neural network, with the simplest structure being a feedforward network. In a feedforward network, signals flow only in one direction, since there are no backward activities in the network.

The most basic form of a feedforward neural network is the perceptron. A key feature of the perceptron is that it registers only two possible results: “1” and “0”. The value of “1” triggers the activation function, while the value of “0” does not.

In supervised learning, perceptrons can be used to train data and develop a prediction model. The steps for training data are as follows:

- Inputs are fed into the processor (neurons / nodes).
- The perceptron estimates the weights of these inputs.
- The perceptron calculates the error between the estimate and the actual value.
- The perceptron adjusts its weights according to the error.

The above four steps are repeated until the desired accuracy of the model is reached. The shortcoming of the perceptron is that, since the output signal is binary (1 or 0), small changes in weights or bias in any single perceptron in a larger neural network can cause polarizing results. This can lead to significant variations in the network and a radical change of the final result.

An alternative to the perceptron is the sigmoid neuron. A sigmoid neuron is very similar to a perceptron, but having a sigmoid function, rather than a binary model, allows accepting any value between 0 and 1. This provides more flexibility to absorb small changes in edge weight without
triggering reverse results since the output is no longer binary. In other words, the output result will not be changed radically just because of one minor change in the edge weight or input value.

In recent years, there has been a strong interest in deep learning. What makes deep learning “deep” is the stacking of at least 5-10 node levels with advanced object recognition using more than 150 layers [8, 9].

Object recognition, used in self-driving cars to recognize objects such as pedestrians and other vehicles, is a popular deep learning application today. Other common deep learning applications include time series analysis and forecasting to analyze data trends measured over specific time periods or intervals, speech recognition, and text processing tasks [10, 11].

Multi-layer perceptrons have been largely replaced by new deep learning methods such as convolution networks, recurrent networks, deep belief networks, and recursive neural tensor networks (RNTN) [12, 13]. These more advanced iterations of the neural network can be effectively used in a number of practical applications that are widespread currently. While convolution networks may be the most popular and powerful of deep learning methods, new techniques and possibilities are constantly evolving [14, 15].

3. A neural network model

A neural network is a mathematical model consisting of a set of interconnected elements. In general, the working principle of a neuron can be written in two equations (see description of figure 1).

\[ v_k = b_k + \sum_{i=1}^{m} w_{ki}x_i = \sum_{i=0}^{m} w_{ki}x_i \, , \text{where} \, w_{k0} = b_k, x_0 = 1 \]

\[ y_k = \phi(v_k) \]

\[ \sum \]

\[ \begin{cases} 1, & v_k \geq 0 \\ 0, & v_k < 0 \end{cases} \]

This has been the first activation function introduced; it is described in the work of McCulloch and Pitts.

\[ \phi(v_k) = \begin{cases} 1, & v_k \geq a \\ v_k, & -a \leq v_k < a \\ -1, & v_k \leq -a \end{cases} \]

where \( a \) – some threshold value. Here, the function \( \phi(v_k) = v_k \) is selected as the linear part, but any other linear function with different coefficients can be used instead.

Figure 1. Artificial neuron diagram.

Any function can be used as an activation function. Let us list the ones that are most often used when constructing a neural network:

- Threshold activation function (2):

\[ \phi(v_k) = \begin{cases} 1, & v_k \geq 0 \\ 0, & v_k < 0 \end{cases} \]

- Piecewise linear activation function (3):

\[ \phi(v_k) = \begin{cases} 1, & v_k \geq a \\ v_k, & -a \leq v_k < a \\ -1, & v_k \leq -a \end{cases} \]
• Sigmoid activation function (4):

\[ \varphi(v_k) = \frac{1}{1 + e^{-av_k}} \tag{4} \]

where \( a \) – the parameter that determines the slope of the function. Often, instead of a sigmoid function, a hyperbolic tangent with the same properties is used (5).

\[ \varphi(v_k) = th(v_k) = \frac{e^{v_k} - e^{-v_k}}{e^{v_k} + e^{-v_k}} \tag{5} \]

The sigmoid function and the hyperbolic tangent activation function are the most popular at present. From a mathematical point of view, a neural network is a multiple superposition of polynomial sigmoid-like functions, which are graphically represented in Figure 2.

\[ F = k_0 \zeta \left( k_1 \zeta \left( \sum_{i=1}^{m} k_2 \zeta \left( \sum_{j=1}^{n} k_3 \zeta (k_4 x_{ij}) \right) \right) + k_5 \zeta \left( \sum_{i=1}^{m} k_6 \zeta \left( \sum_{j=1}^{n} k_7 \zeta (k_8 x_{ij}) \right) \right) + \ldots \right) \tag{6} \]

![Figure 2. Neural network.](image)

The input and the output coefficients of neurons, which determine both the type of dependence (input) and the effect of the neuron (output) on other neurons of the network, give a special character to the function of the neural network. The selection of these coefficients is called neural network training.

There are three types of training.

The stochastic method (introduced by T. Kohonen) involves the enumeration of random values of the coefficients until the function of the neural network adequately reflects the desired dependence. The disadvantage of this method is the low network learning rate.

The gradient method (backpropagation of error) involves changing the coefficients of the network with respect to the error gradient of the sigmoid function computed through the derivatives, in such a way as to minimize the error. The disadvantage of the method is that it is impossible to find alternative solutions when the error minimum is reached in cases when the function minimum is not global. This peculiarity is known as falling into the local minimum trap.

Mixed methods presume the simultaneous use of both stochastic and gradient components of learning [16]. An example of such a combination is the evolutionary algorithm used in the MS Excel Solver module.

4. Neural network modelling in economics and finance

Let us consider an autocorrelation prediction for one period for the USD/RUB exchange rate on a single-layer neural network [17, 18]. Let us calculate the formula for the network output (see figure 3). The output deviation is obtained as the square of the difference between the output of the neuron and the actual value (see figure 4).
In this way, we obtain the forecast based on the neural network (see figure 5):

5. Deep neural networks
Networks with a large number (hundreds of thousands) of neurons, (tens and hundreds) of hidden layers and complicated, usually specialized, architecture are referred to as deep neural networks. Training such networks requires considerable time and a large set of both correct and erroneous examples of solutions. However, the performance of such networks is high enough, which allows them to be used in serious scientific and commercial projects.

Specialized deep learning algorithms are various modifications of the mixed method, aimed at the parallel execution on a large number of neurons or a group of neural networks. For training and operation of this type of neural networks, neural processors (Neural Processing Unit, NPU) are used – specialized equipment designed solely for the task of mathematical modelling of neural networks.

The most popular commercial neural processors are listed below.
NVIDIA DGX-2 – 8,2×104 neurons, 1.28×109 coefficients, 2×1015 FLOPS, dimensions: 52×26×64 cm, power consumption: 103 W, price: $ 400,000. Turn-key computational system for network training.
IBM TrueNorth – Simultaneous processing: 106 neurons, 2,56×108 coefficients, 4,5×1012 FLOPS, dimensions: 2×2×1 cm, power consumption: 0,1 W. Turn-key customized neural servers.

Nvidia Tesla M4 – Simultaneous processing: 1024 neurons, 2048 coefficients, 2,2×1012 FLOPS, dimensions: 17×8×5 cm, power consumption: 75 W.

Apple A12 Bionic – Simultaneous processing: 8 neurons, 16 coefficients, 5×1012 FLOPS, dimensions: 3×3×0,5 cm, power consumption: 6 W. On-a-chip neural module in the iPhone XS.

6. Conclusion
The advantages of using neural networks include: versatility, simplicity. Neural networks are able to model dependencies also in cases when there is a large number of variables.

Yet, besides the positive aspects of neural network models, they have significant drawbacks:

- Complexity of constructing the network architecture for a specific task. For the vast majority of real-world problems, there are no standard schemes, and thus, in each case, the construction has to start “from scratch”.
- Difficulty in interpreting learning outcomes. A neural network involves multiple nonlinear transformations and, notably, the more layers the network has, the more difficult it is to interpret these transformations.

References
[1] Qiu M, Song Y and Akagi F 2016 Application of artificial neural network for the prediction of stock market returns: The case of the Japanese stock market Chaos, Solitons and Fractals 85 1-7
[2] Radosteva M, Soloviev V, Ivanyuk V and Tsvirkun A 2018 Use of neural network models in the market risk management Advances in Systems Science and Applications 18(2) 53-58
[3] Lahmri S and Bekiros S 2019 Cryptocurrency forecasting with deep learning chaotic neural networks Chaos, Solitons & Fractals 118 35-40
[4] Okwuichi I 2020 Machine Learning based Models for Fresh Produce Yield and Price Forecasting for Strawberry Fruit Master's thesis, University of Waterloo
[5] Elizarov M, Ivanyuk V, Soloviev V and Tsvirkun A 2017 Identification of high-frequency traders using fuzzy logic methods. In 2017 Tenth International Conference Management of Large-Scale System Development MLSD IEEE pp. 1-4
[6] Koroteev M V, Terelyanskii P V and Ivanyuk V A 2016 Fuzzy Inference as a Generalization of the Bayesian Inference Journal of mathematical sciences 216(5) 685-691
[7] Koroteev M V, Terelyanskii P V and Ivanyuk V A 2016 Arithmetic of fuzzy numbers in generalized trapezoidal form Journal of Mathematical Sciences 216(5) 696-701
[8] Gallicchio C and Micheli A 2017 Echo state property of deep reservoir computing networks. Cognitive Computation 9(3) 337-350
[9] Shi H, Xu M and Li R 2017 Deep learning for household load forecasting—A novel pooling deep RNN IEEE Transactions on Smart Grid 9(5) 5271-80
[10] Wang H, Lei Z, Zhang X, Zhou B and Peng J 2019 A review of deep learning for renewable energy forecasting Energy Conversion and Management 198 111799
[11] Wang H Z, Li G Q, Wang G B, Peng J C, Jiang H and Liu Y T 2017 Deep learning based ensemble approach for probabilistic wind power forecasting Applied energy 188 56-70
[12] Kong W, Dong Z Y, Jia Y, Hill D J, Xu Y and Zhang Y 2017 Short-term residential load forecasting based on LSTM recurrent neural network IEEE Transactions on Smart Grid 10(1) 841-851
[13] Simonov B V, Ivanyuk V A and Simonova I E 2015 Existence of best approximation elements in the spaces $L_{0+\phi}$−. Contemporary Mathematics and Its Applications 96 112-131
[14] Dheir I M, Mettleq A S, Elsharif A A and Abu-Naser S S 2020 Classifying Nuts Types Using Convolutional Neural Network
[15] Li Y, Yu R, Shahabi C and Liu Y 2017 Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926
[16] Taormina R and Chau K W 2015 Neural network river forecasting with multi-objective fully informed particle swarm optimization. Journal of Hydroinformatics 17(1) 99-113
[17] Ivanyuk V 2018 Econometric Forecasting Models Based on Forecast Combination Methods. In 2018 Eleventh International Conference" Management of large-scale system development" MLSD IEEE pp. 1-4
[18] Qiu X, Zhang L, Ren Y, Suganthan P N and Amaratunga G 2014 Ensemble deep learning for regression and time series forecasting In 2014 IEEE symposium on computational intelligence in ensemble learning (CIEL) IEEE pp. 1-6