Weather Forecasting System with the use of Neural Network and Backpropagation Algorithm

Marek Wica, Mirosław Witkowski, Anita Szumiec, Tomasz Ziebura
Faculty of Applied Mathematics
Silesian University of Technology
Kaszubska 23, 44-100 Gliwice, Poland
Email: anitszu502@student.polsl.pl

***Abstract*—People are searching effective ways for weather prediction for ages. Knowledge about atmospheric phenomenon might become helpful in predicting and preventing disasters in the future. In recent years weather prediction methods are more and more accurate. Technological development has helped humanity in many areas of life. It is not different when it comes to weather forecasts. In this article we featured an application which determines forecast of temperature and amount of rainfall for the next day when it is provided certain input data from past days. This is made possible by neural networks and backpropagation algorithm. In the next section of the article aforementioned algorithms and the way we used them will be described as well as the end result. Our system is created using Python language**

I. Introduction

Recent advances in artificial intelligence show many novel applications of neural networks. These structures simulate thinking processes so they are applicable in various engineering and computational exercises. In [1]–[3] neural networks served as predictive models for electrical production and dispatchment. There are also many advanced architectures of neural networks which are used for verification over biometrics or voice [4]. Neural networks are also very efficient simulators of dynamic processes over time interval and various initial conditions [5]. Other implementations of neural networks show their efficiency in smart control of home environments [6]. Therefore in recent time we can find various approaches to use neural networks and other artificial intelligence methods for weather forecast. In [7] a Markovian process model was implemented to serve as a support for weather prediction over a short time interval. Computational methods serve also as a support in agriculture weather prediction, ie. in rice farming [8]. While in [9] neural networks for weather forecast were trained by the use of fine tuning. There are many interesting approaches to use artificial intelligence in weather prediction, while a comprehensive summary of these research can be found in [10]. Our model to predict rain is based on neural networks and as data for training we have used information from ground weather stations in our region.

Application which we have developed was created on the basis of weather data downloaded from meteorological website https://www.wunderground.com. Data we used: date, average temperature, average dew temperature, maximum humidity, minimum humidity, maximum temperature, minimum temperature, maximum dew temperature, minimum dew temperature, maximum pressure, minimum pressure, precipitation(rainfall). Aforementioned data comes from years 2015-2017. For the proper operation of the application aforementioned data for 3 past days is needed in order to determine the next day. In order to make the data readable to the application first we had to convert spreadsheets into JSON format, the next operation was to create dataframe from json array in order to derive necessary data, prepare and clean the dataset lastly save it to a easy readable file.

Project is in the form of a website. Backend was developed with Django, as for the frontend layer we have used simple template created with HTML5 and CSS3. Django is open-source Web Framework. It is base on Python Language. Django implements the model-template-view (MTV) architectural pattern. There are a lot of forecasting systems, because weather applies to each of us. Every big TV station have own forecasting system and shows result on tv channel. The most interesting examples: Global Forecast System (GFS), The Weather Channel, WeatherPro, MeteoICM. Furthermore in our country is in use Regional Warning System, which is a free service warning citizens living in specific municipality against risk. By term "weather forecasting" we understand to predict values of minimum, average and maximum temperature and amount of precipitation for a given day based on date of preceding 3 days. To do this, we create a neural network using a back propagation algorithm.

II. Description of entire system

1. First we splitted the dataset into two separate sets: training set and testing set. After the creation of input vector the neural network is calculating the output vector and based on it and the training set the error is being computed.
2. The error was propagated successively from the output layer to the first layer via the backpropagation algorithm. Based on this error weights of the connections between neurons were adjusted for obtaining more realistic results.
3. This procedure was repeated to minimize the error. As an input vector further data from training set is provided.
4. After the calculation of weights for the entire training set for hundreds of times we received computed weights.

©2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-2468/p8.pdf

To determine the effectiveness of these computed weights our neural network has calculated values of output vector for subsequent input vectors from the testing set.

5. Data from the testing set cannot be used for training the neural network. In this set there have to be input as well as output data.
6. During the test set calculations we were computing the error on the output layer compared to the data from testing set after each and every input vector calculation. As a result we received chart of the errors and after setting a permitted error threshold we could determine the effectiveness of our neural network.

III. Neural network structure

We created class Neuron and Synapsa in order to represent individual neurons and connections between them. Every connection have own weight, which will be used during calculating. Our neural network consist of one input layer, one output layer and two hidden layers. To input layer are given 33 values: average temperature, average dew temperature, maximum humidity, minimum humidity, maximum temperature, minimum temperature, maximum dew temperature, minimum dew temperature, maximum pressure, minimum pressure, precipitation(rainfall) for preceding 3 days. On the output layer we received 4 values: minimum, average, maximum temperature and amount of precipitation. We have two hidden layer.

First hidden layer have 11 neurons, because we have 11 values for one day and on input layer we give data for three days. Second layer have 5 neurons based on characterisation of individuals data and what they are about.

To compute input value on individual neuron, we added values entering neuron. The following formula illustrate this.

$$y_{in} = \sum_{i=1}^{n} w_i * x_i \tag{1}$$

On the output of individual neuron, we transferred the input value to activation function. So, the formula look like this.

$$y_{out} = f(\sum_{i=1}^{n} w_i * x_i) \tag{2}$$

Activation function will be presented in the next following section.

IV. Activation function

This term is used in artificial intelligent, more particularly in neural networks. Activation function is used to calculate the output value of individual neuron. We used unipolar sigmoid function, because it seems the most optimal. It function is continuous, reactive and it reacts for changing input values. Parameter was chosen based on experience while substituting various values. Formula of activation function:

$$f(x) = \frac{1}{1 + \exp(-a * x))} \tag{3}$$

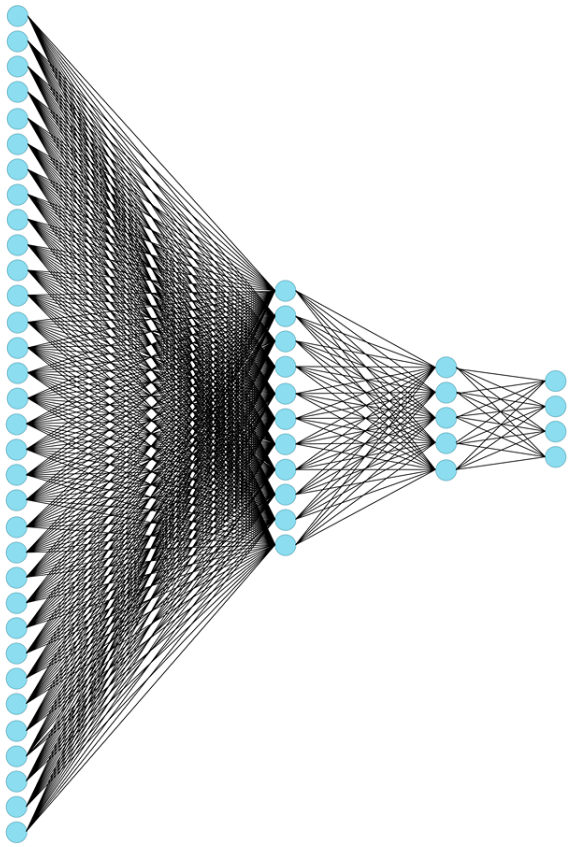

Figure 1: Structure of Neural Network

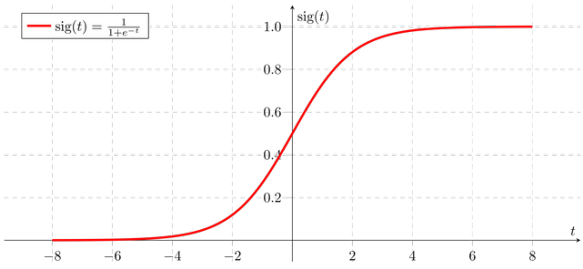

Figure 2: Graph of the Activation Function

On the next steps we will use derivative of the activation function.The derivative of the activation function is in the following form.

$$f'(x) = \frac{a * \exp(-a * x)}{(\exp(-a * x) + 1)^2} \tag{4}$$

V. Error function

In order to determine the effectiveness of our neural network model we have used medium square error. The better the design of the network model, the smaller the medium square error was. Error was calculated on the output layer on individual neurons. Then, was calculated error of the entire

network. We used the following formula for Mean Squared Error:

$$\frac{1}{n} * \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{5}$$

This error was propagated to the previous layers using Back Propagation Algorithm.

VI. Network learning and Backpropagation Algorithm

Neural network learning is based on correction weight of connection between neurons. New value of weight is equal to sum of old weight and delta value of weight.

$$w_n = w_o + \Delta w \tag{6}$$

Whereas delta value of weight is equal to negated partial derivative of error function with respect to the value of individual weight multiply by constant value of learning rate. Learning learn parameter was chosen based on experience while substituting various values. It is usually in range of 0.125 to 0.5.

$$\Delta w = -\eta \frac{\gamma E}{\gamma w} \tag{7}$$

After the derivative is extended, we can see, that delta value is equal to multiplication of learning rate, input value of individual neuron, derivation of activation function and difference between predicted value and received(calculated) value.

$$\Delta w = \eta * (o - y) f'(z) * x \tag{8}$$

That approach is known as Delta Rule. It is sort of gradient descent learning method. However, this rule can be applied only to single-layer neural network. Our neural network have more layers. In our learning set we have predicted values on the output layer, but we have not output values of individual neurons in hidden layers. It was a huge issue for scientists, who worked on development of neural network to correct weight in hidden layers. One day, was invented Backpropagation Algorithm, which allows to correct weight in hidden layer, without knowledge of output values of individual neurons in hidden layers. This algorithm is more versatile.

So, we used more general algorithm know as Backpropagation Algorithm. The main idea is the same, to correct weight through propagate error to previous layers. We assume that multiplication of derivation of activation function and difference between predicted value and received(calculated) value from the previous formula will be represent as lower delta.

$$\delta = (o - y) * f'(z) \tag{9}$$

And this value of lower delta is applied to last, output layer. For other layers, lower delta be equal to weighted sum of lower delta from next layer multiplied by derivation of activation function.

$$\delta j = f'(z) * \sum_{k=1}^{K} \delta_{(j+1)k} * w_{(j+1)k} \tag{10}$$

j - currently analyzed layer of neurons
(j + 1) - next layer of neurons
K - amount of neurons in the layer (j + 1)
k - individual neuron from the layer (j + 1)

In this way, further weights are correcting and it is how Backpropagation Algorithm works.

VII. Normalization and denormalization

Maybe You notice that, our data set consist of values from different ranges. However, our activation function returns values from range 0 to 1. To work it correctly, we have to apply process, known as Normalization, which allows to comparison data each other and to further analyze. We used min-max normalization, which transforms data to range 0 to 1. The following formula shows min-max normalization.

$$V' = \frac{V - min}{max - min} * (new_{max} - new_{min}) + new_{min} \tag{11}$$

In our case new min is equal to 0 and new max is equal to 1.

In order to read the data in a conveniently way, we applied reverse process of normalization, known as denormalization to read values on the output layer during predicting process. Formula for denormalization, was derived from normalization formula and it is as follow.

$$V = (V' * (max - min) + min) \tag{12}$$

VIII. Graphical User Interface (GUI)

The GUI which was used to create user friendly interface for out web application was developed using Django as backend framework and simple HTML5 and CSS3 template as the frontend layer for posting the form's data. Django is a free backend framework written in python. It implements model-template-view architectural model which guarantees code clarity and simplicity. We have chosen these technologies because it ensures security, maintainability, effectiveness as well as compatibility and easiness of implementing our neural network since both were written in python.

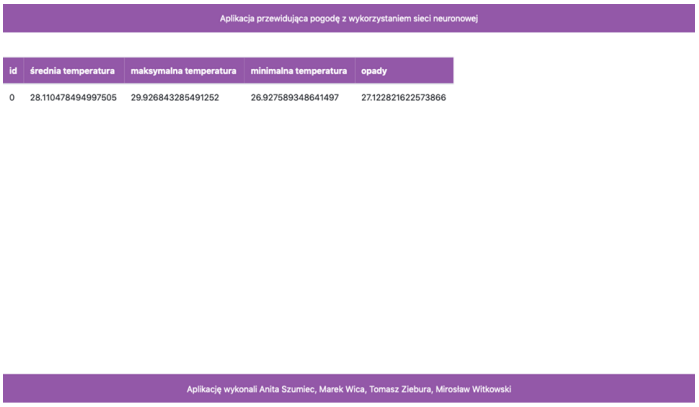

Figure 3: The Appearance of the GUI

IX. Preparing data sets

Data set was prepared based on the historical data of the website https://www.wunderground.com, first we prepared spreadsheets for each year begging on the year 2015 and ending on year 2017. Next we have converted each day contained in the spreadsheets into an object and created array containing every day. Lastly we have exported the array in a .json file. Our DailySummary object was based on: date, average temperature, average dew temperature, maximum and minimum humidity, maximum and minimum temperature, maximum and minimum dew temperature, maximum and minimum pressure, and precipitation(rainfall). Based on these parameters for last 3 days the weather was predicted for the next day.

X. Results of learning

As was mentioned, we had three years of weather data for specified locations. It location was Katowice, Ożarowice in Poland. We had circa 1000 data vectors (records). Only learning set was over 700 data vectors. If we take into account that every vector was calculated through neural network and back using backpropagation algorithm circa thousand times(what is called epoch), we can imagine how much calculations computer had to do. Learning process has taken a few hours. After every change of weight, they was saved to the 3-dimensional array. After converting all epochs, This 3-dimensional array was saved to the file. This file was imported by neural network, which was build in predicting mode(without backpropagation algorithm). During learning process, also error made was saved to the array, after each epoch. After converting all epochs, we had array of errors. That way we could see, how error was reducing during learning process. Now, we show it on the graph. It is how error correction was looking. Curve illustrates how error was changing.

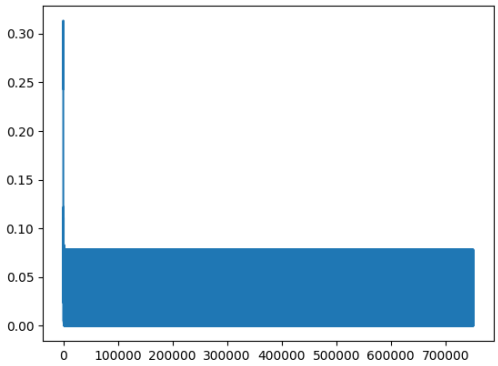

Figure 4: Graph of Error Correction

It looks very interesting. When we zoom this chart.

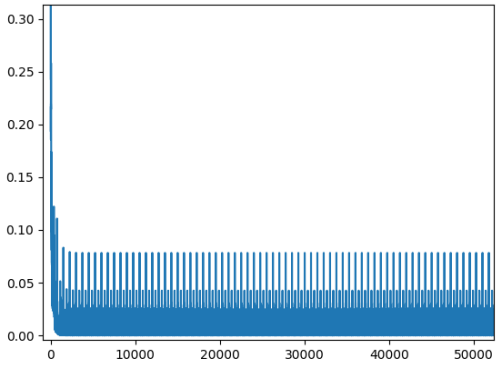

Figure 5: Zoomed Graph of Error Correction 1

We can see, that there are a lot of peaks. On the start, error was high and during learning process, curve, which illustrates errors, was descending, until it reaches certain point. After that, we have a lot of peaks. We zoom it, to see it better.

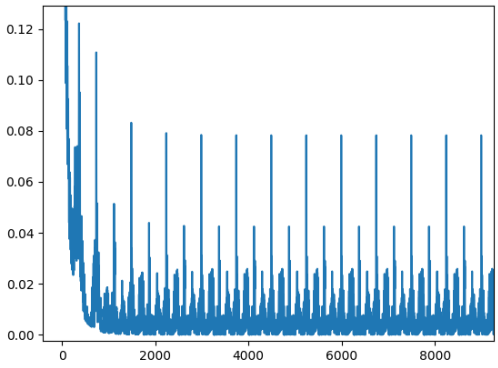

Figure 6: Zoomed Graph of Error Correction 2

After that certain point, our error function probably, fell into the local minimum.

XI. Testing

When all calculations was done, we got file with weights. Before we used this file to neural network in predicting mode, we had tested these weights on neural network in testing mode. In testing mode we used testing set with circa 350 data vectors. After calculating each data vector, error was calculated, compared to predicted values and saved to the array. This allowed us to make a graph with these values. So this is how it looks like.

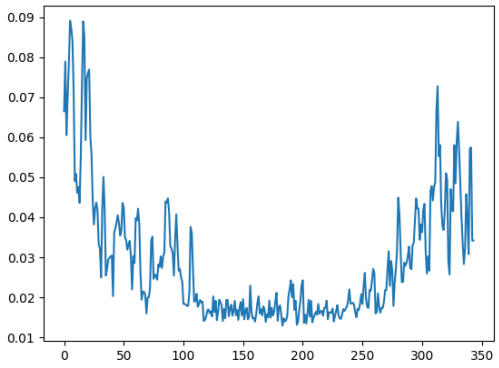

Figure 7: Graph of Error during Testing Process

The graph depicts, that at the beginning of testing process, error is the biggest. Later the error curve drops, but at the end, curve goes up. We can conclude that, neural network will better predict weather for certain season. When we accept the error of the order of 0.07, efficiency of our neural network will equal to 95.93 percent, that is excellent result. Moreover when we run our neural network in predicting mode on the live test, our system correctly predict weather for day of the system start. It was really rain on that day!

XII. Conclusions

In summary, we build neural network with backpropagation algorithm. We based on the three years of data. It was enough to reach excellent results, sufficiency correct error and correctly predict weather for certain day. Results of learning and testing was presented also on graphs in order to better illustrate it. With a good designed structure of neural network, we can reach high efficiency with low error level.

Great unipolar sigmoid function, which was chosen as activation function, helped us. It function is very good, because it is continuous, reactive and it reacts for changing input values. Due to the use this function, we had to use normalization and denormalization. It is great example, how knowledge of programming, web development technologies, mathematics, calculus, statistics and appropriate algorithms, can helped us to build system, which will make our world better and will save our life before disasters.

References

[1] S. Brusca, G. Capizzi, G. Lo Sciuto, and G. Susi, "A new design methodology to predict wind farm energy production by means of a spiking neural network–based system," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 32, no. 4, p. e2267, 2019.

[2] S. Coco, A. Laudani, F. R. Fulginei, and A. Salvini, "A new neural predictor for elf magnetic field strength," *IEEE Transactions on Magnetics*, vol. 50, no. 2, pp. 69–72, 2014.

[3] F. Bonanno, G. Capizzi, G. L. Sciuto, and C. Napoli, "Wavelet recurrent neural network with semi-parametric input data preprocessing for micro-wind power forecasting in integrated generation systems," pp. 602–609, 2015.

[4] D. Połap, M. Woźniak, R. Damaševičius, and R. Maskeliūnas, "Bio-inspired voice evaluation mechanism," *Applied Soft Computing*, vol. 80, pp. 342–357, 2019.

[5] M. Woźniak and D. Połap, "Hybrid neuro-heuristic methodology for simulation and control of dynamic systems over time interval," *Neural Networks*, vol. 93, pp. 45–56, 2017.

[6] ——, "Intelligent home systems for ubiquitous user support by using neural networks and rule based approach," *IEEE Transactions on Industrial Informatics*, 2019.

[7] S. Kaneriya, S. Tanwar, S. Buddhadev, J. P. Verma, S. Tyagi, N. Kumar, and S. Misra, "A range-based approach for long-term forecast of weather using probabilistic markov model," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2018, pp. 1–6.

[8] J. Cao, J. Tan, Y. Cui, and Y. Luo, "Irrigation scheduling of paddy rice using short-term weather forecast data," *Agricultural water management*, vol. 213, pp. 714–723, 2019.

[9] S.-Y. Lin, C.-C. Chiang, J.-B. Li, Z.-S. Hung, and K.-M. Chao, "Dynamic fine-tuning stacked auto-encoder neural network for weather forecast," *Future Generation Computer Systems*, vol. 89, pp. 446–454, 2018.

[10] R. B. Alley, K. A. Emanuel, and F. Zhang, "Advances in weather prediction," *Science*, vol. 363, no. 6425, pp. 342–344, 2019.