Water Potability Classification using Neural Networks

Patryk Rozynek[1], Michal Rozynek[1]

[1]*Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, POLAND*

Abstract

Nowadays, the Internet of Things, intelligent systems are becoming very popular and wanted. These tools can be used for smart and fast analysis of different data. In this paper, we focused on the automatic analysis of water quality by the use of artificial intelligence methods. As the main tool for this analysis, k-nearest neighbor and artificial neural network were used. Both methods were tested, and the results were discussed in terms of selecting the best tool in the topic of water potability.

Keywords

water potability, artificial neural network, kNN algorithm

1. Introduction

Today, the Internet of Things (IoT) is one of the most important areas of developing and practical implementing smart solutions. Especially, the last years show that smart things can be used everything for different management and systems. One such area is water management and automatic evaluation/analysis. In [1], a system for water management was shown where sustainable networks were analyzed. Moreover, similar solutions were shown in [2, 11]. The authors of this research use deep machine learning like a convolutional neural network for the analysis of water pollination for agricultural irrigation resources. Not only in water management, machine learning solutions are used, but also to detect and classify different objects on water. One such task is to analyze the ship passing some areas. In [3, 4, 9], two solutions for taking an image on a river and used for classification purposes were presented. Both solutions show practical potential in implementation based on performed real case studies. All machine learning solutions in IoT solutions uses a whole data [5, 6, 7, 8, 10] to analysis or extracted features [12, 13, 14, 15]. In both cases, the results show great accuracy in using these approaches. In this paper, we propose a solution for analyzing the water by the use of two tools like K-nn and an artificial neural network.

2. Methodology

The algorithm classifies whether the water with the given properties is safe to drink. We used two classification methods: the KNN algorithm. The code of the algorithms were written in Python. Most of the article compares these algorithms to see which one is more suitable for use in our project. The project was implemented using the database available on the website https://www.kaggle.com/. The algorithm is based on counting the distance between the given sample and each object in the training set. We used the Minkowsky algorithm to calculate the distance:

$$D(x,y) = (\sum_{i=1}^{m} |x_i - y_i|^r)^{1/r} \tag{1}$$

In this way, we obtain a table of distances from our sample. We just need to sort it so that the shortest distances are at the front. The classification decision is then made by voting 'k' neighbors or 'k' of the first objects in the distance table. The result will be the value of the class that was voted more times. If there are the same number of votes for both variants of the class, then the algorithm chooses the first of them and votes for it. Therefore, it is recommended that the number of neighbors is odd.

3. Artificial neural network

A neural network is a software modeled after the operation of neurons in the human brain. They consist of three types of layers: input (it collects data and passes it on), hidden (here the connections between neurons are looked for, here the learning process takes place), and output (collects conclusions, analysis results). Typically, the neural network is made up of many layers. The first layer - as in the case of images recorded, for example, by the optic nerves of a human - goes to raw input data. Each subsequent layer receives data as a result of data processing in the previous layer. What the last layer produces is the so-called System output. A neural network functions like the human brain: each neuron carries out its simple calculations, and the network made up of all neurons multiplies the potential of that calculation. Neural networks used in artificial intelligence are organized on the same principle - but with one exception: to perform a specific task, the connections between neurons can be adjusted accordingly. Searching for information

ICYRIME 2021 @ International Conference of Yearly Reports on Informatics Mathematics and Engineering, online, July 9, 2021

patrroz599@student.polsl.pl (P. Rozynek);
patrroz599@student.polsl.pl (M. Rozynek)

```
1  class KNN:
2      def clustering(testSample,y,k,classes):
3          x = y.copy()
4          dist=[]
5          for i in range(len(x)):
6              dist.append(KNN.minkowsky(testSample,x.iloc[i],
                   2))
7          KNN.sort(dist,x)
8          for i in range(0,k,1):
9              classes[x.iloc[i][9]]+=1
10         return max(classes,key=classes.get)
```

is the process of searching a specific set of documents relating to the subject or object indicated in the query or containing facts necessary for the user. However, this process has not been precisely and finitely defined by patterns, standards, or algorithms and is largely based on heuristics, in this case, defined as a set of rules and guidelines that may or may not lead to the right solution. For each neuron, the sum of the products of previous neurons and associated synapses (weights) is calculated. The result is then passed on to the activation function. The formula for calculating the value of a neuron:

$$o = f(\sum_{i=0}^{n} x_i * w_i) \tag{2}$$

where o - output, w - weights, x - neurons The activation function can be any function. It is often taken as a hyperbolic tangent. After all the values for the neurons in the hidden layer have been calculated, the output layer is recalculated in the same way. This layer has as many neurons as there are classes. Then the global error is calculated based on the values from this layer. If the error is smaller, the weights that were used are saved and the previous ones are forgotten. Heuristics is a method of finding solutions for which there is no guarantee of finding the optimal, or often even correct, solution. These solutions are used, for example, when the full algorithm is too expensive for technical reasons or when it is unknown. The method is also used to find approximate solutions, based on which the final result is calculated using the full algorithm. The latter application primarily applies to cases where heuristics are used to direct the full algorithm to the optimal solution to reduce the program runtime in a typical case without sacrificing the quality of the solution.

4. Description

Pseudocode: Part of code in Python: (1)

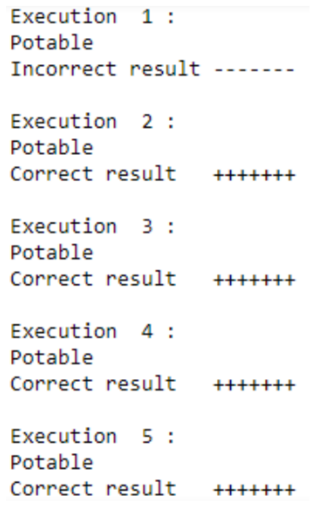
- 2 - 3 (1) After getting the input data, the training set is copied to the new facility

Algorithm 1: KNN Algorithm

```
Data: Input: Sample list of features, training set x,
      number of neighbors k, possible classes
      classes
Result: The result of the vote
1  $dist := []$;
2  $i := 0$;
3  while $i < length(x)$ do
4      $distance := 0$;
5      $j := 1$;
6      while $j < length(Sample)$ do
7          $distance$+=Minkowsky(sample[j], x[i][j]);
8          $j$+=1;
9      end
10     add to list $dist$ value of $distance$;
11     $i$+=1;
12 end
13 sort the list $dist$;
14 do the same replacements for the list $x$;
15 $n := 0$;
16 while $n < k$ do
17     vote for the class he has $n - th$ list component
       $x$;
18     $n$+=1;
19 end
20 Return the class with the most votes;
```

```
1  def minkowsky(v1,v2,m):
2      distance=0
3      for i in range(len(v1)-1):
4          distance+=abs(v2[i]-v1[i])**m
5      distance=distance**(1/m)
6      return distance
```

- 4 (1) A distance table is created for the training set
- 5 - 6 (1) Minkowsky is called for each sample in the training set. The first argument is an example set of water parameters, and the second is another sample from the training set.
- 3 - 5 (2) The distance is counted.
- 6 (2) The distance between the sample sample and the comparison object to the distance table is returned.
- 7 (1) The algorithm sorts the list of distances so that the most similar cases are at the top of the list. The same changes are performed on the training set
- 7 8 - 9 (1) Now is the vote. The kof the first records on the distance list are taken. Each neighbor votes for the class they own.
- 10 (1) The class with the most votes is returned. If they have tied votes, the first class, Potable is



```
Execution  1 :
Potable
Incorrect result -------

Execution  2 :
Potable
Correct result   +++++++

Execution  3 :
Potable
Correct result   +++++++

Execution  4 :
Potable
Correct result   +++++++

Execution  5 :
Potable
Correct result   +++++++
```

Figure 1: An example of the algorithm execution during the research

returned.

Sample program execution is shown in Fig.1:

5. Experiments

5.1. KNN

The KNN algorithm achieved the following results presented in Fig. 2. Based on Fig. 2, the accuracy of the performed method shows very similar results. For analyzed three different numbers of neighbors like k $\in \{1, 2, 3\}$ the accuracy was on the same level that is 50%. We can say that the number of neighbors (on a small number of parameters k is irrelevant for the classification task. But more importantly, why is the effectiveness so low? The base we used has drinking and non-drinking water records with similar parameters. For example, the figure below shows potable and non-potable water samples in Fig. 3. As you can see, an example feature has different values for potable and non-potable water. The next example in the picture in Fig. 4.

The Hardness trait for drinking water has a wider range of values, but the average values are very close to each other. To sum up, the use of knn algorithm seems to be a tool that for a small number of parameters k the results are not promising. The accuracy on a level of 50% cannot be used in practical implementation.

5.2. Neural Network

The set was divided into a training set and a validation set in proportion 1:10. As a result, the program counted

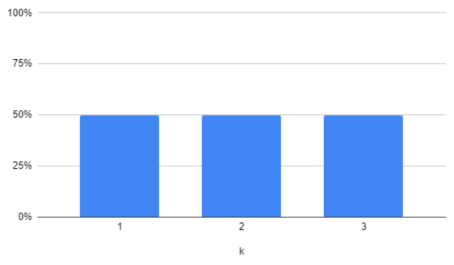


Figure 2: Graph showing the effectiveness of the algorithm depending on the number of neighbors

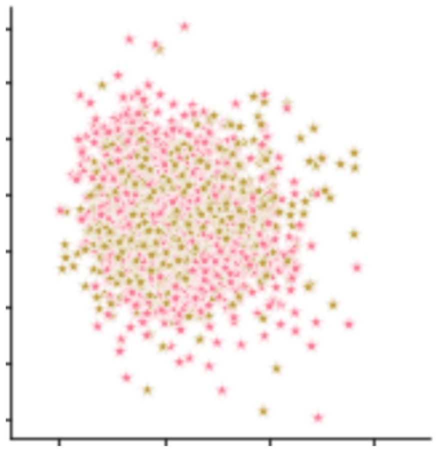


Figure 3: There is no difference between potable and non potable water for an example trait

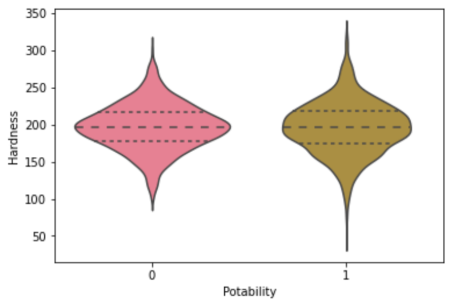


Figure 4: 0 - non-potable water, 1 - potable water, The average values are very similar

faster and the results were almost the same. To check which neural network is the best for our program, we created 15 architectures of artificial neural networks with a different number of hidden layers and the number of neurons in these layers. Three of these nets were deep nets and the rest were shallow. Each network has been trained 1000 times. Additionally, the adopted parameters

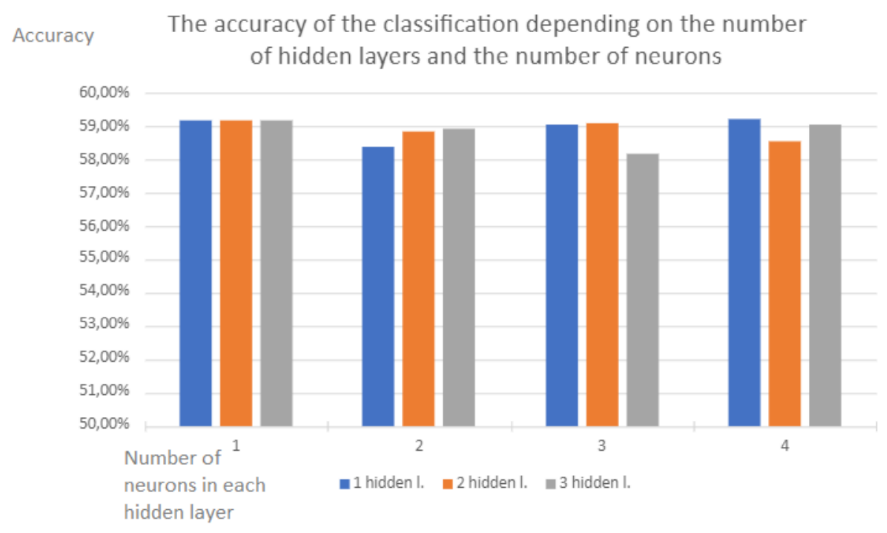


Figure 5: Accuracy

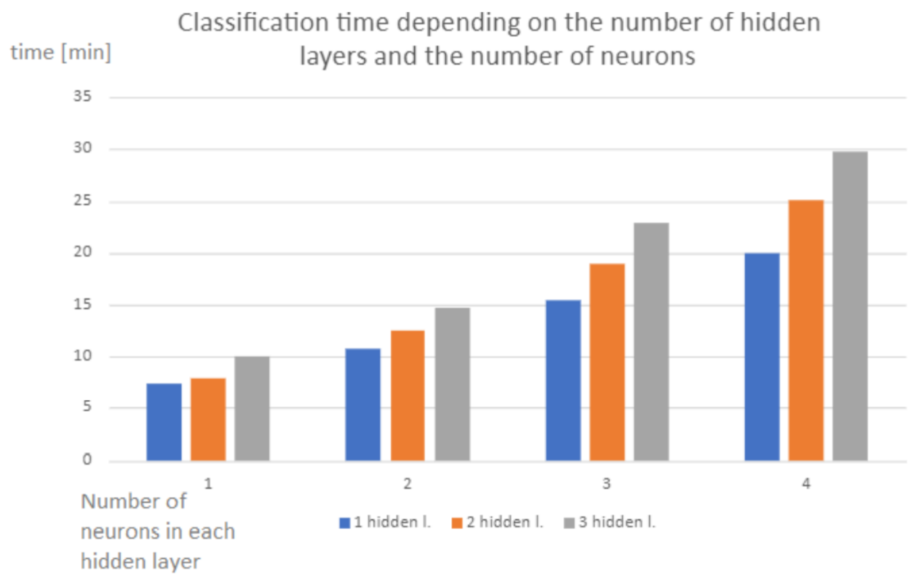


Figure 6: Training Time

were the same for all networks and amounted to: $\phi_1 = 0.1$, $\phi_2 = 0.9$ $\Omega = random =< 0.2; 0.85 >$

The obtained results and comparison for different architectures were presented in Fig.5,6,7 and 8. Based on these results artificial neural network shows much better results then compared knn

6. Conclusion

After performing the tests and comparing the two classification methods, it can be easily stated that using an artificial neural network the results are closer to the truth. An obstacle in the KNN classification was that the values of each of the water properties were too similar. No matter how many neighbors there were, the effect was the same. It can be said that the algorithm guesses the result. No data manipulation, such as deleting one feature, gave

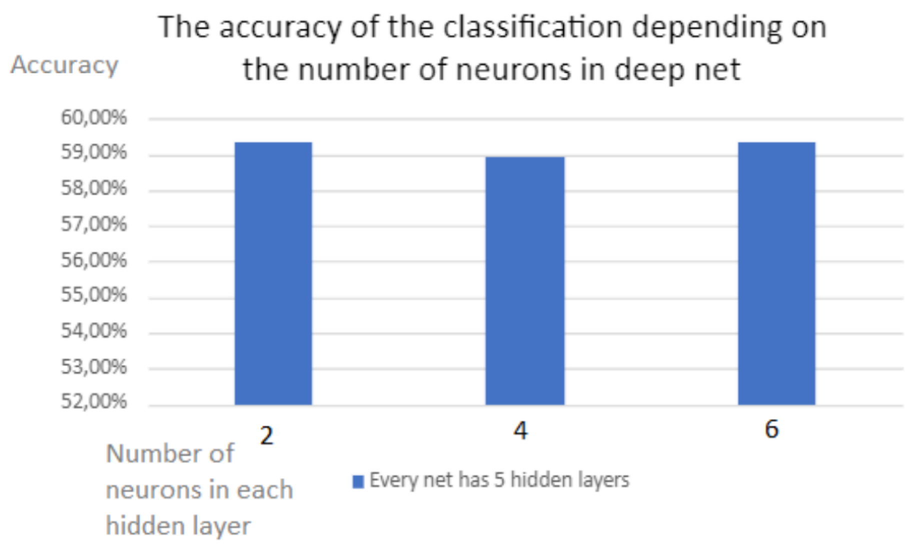


Figure 7: Accuracy

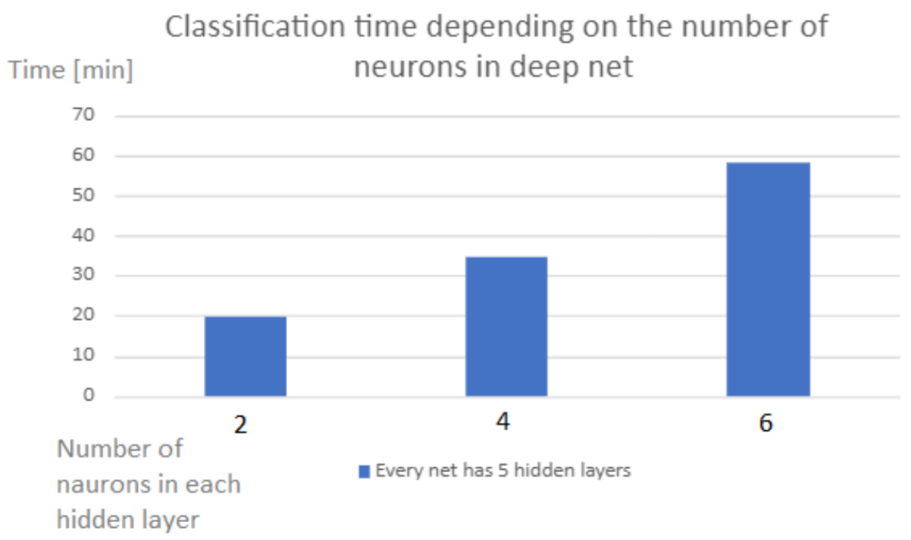


Figure 8: Training Time

better results. In fact, it is also difficult to judge whether the water is drinkable. With the characteristics listed in the database, it is not possible to determine whether the water is potable with the KNN algorithm. When comparing the results of the deep network with the shallow one, it can be said that they are not very different. The two deep nets had the best two accuracies, but the third is below average. Unfortunately to train a network with 5 hidden layers and 6 neurons in each of them took almost an hour. In this type of network, looking at the global error, it changed less frequently. This is because the network has more neurons and therefore more weight. It is more difficult for the network to change the weights so that the next calculated global error is better. Some of the weights may have changed for the better, but the network has not caught them because the error was not any less. The addition of the ability to remove some particles from the swarm made it possible to create new particles with random weights at startup. This is a good way to improve results without investing much in time. Especially if the parameters ϕ_1 and ϕ_2 are 0.1 and 0.2 the particles tend to be the best global particle, not to their best position. The best results are obtained with a network with 5 hidden layers and 2 neurons in this layer and reached 59,32% and took almost 20 minutes

References

[1] H. M. Ramos, A. McNabola, P. A. López-Jiménez, M. Pérez-Sánchez, Smart water management towards future water sustainable networks, Water 12 (2020) 58

[2] H. Chen, A. Chen, L. Xu, H. Xie, H. Qiao, Q. Lin, K. Cai, A deep learning cnn architecture applied in smart near-infrared analysis of water pollution for agricultural irrigation resources, Agricultural Water Management 240 (2020) 106303

[3] D. Połap, M. Włodarczyk-Sielicka, N. Wawrzyniak, Automatic ship classification for a riverside monitoring system using a cascade of artificial intelligence techniques including penalties and rewards, ISA transactions (2021).

[4] T. Hyla, N. Wawrzyniak, Identification of vessels on inland waters using low-quality video streams, in: Proceedings of the 54th Hawaii International Conference on System Sciences, 2021, p. 7269.

[5] V. Nourani, E. Foroumandi, E. Sharghi, D. Dąbrowska, Ecological-environmental quality estimation using remote sensing and combined artificial intelligence techniques, Journal of Hydroinformatics 23 (2021) 47–65.

[6] M. Woźniak, M. Wieczorek, J. Siłka, D. Połap, Body pose prediction based on motion sensor data and recurrent neural network, IEEE Transactions on Industrial Informatics 17 (2020) 2101–2111.

[7] Y. Shao, J. C.-W. Lin, G. Srivastava, D. Guo, H. Zhang, H. Yi, A. Jolfaei, Multi-objective neural evolutionary algorithm for combinatorial optimization problems, IEEE Transactions on Neural Networks and Learning Systems (2021).

[8] C. Napoli, F. Bonanno, G. Capizzi, Exploiting solar wind time series correlation with magnetospheric response by using an hybrid neuro-wavelet approach (2010) Proceedings of the International Astronomical Union, 6 (S274), pp. 156 - 158.

[9] Brociek R., Magistris G.D., Cardia F., Coppa F., Russo S., Contagion Prevention of COVID-19 by means of Touch Detection for Retail Stores (2021) CEUR Workshop Proceedings, 3092, pp. 89 - 94.

[10] G. Capizzi, G. Lo Sciuto, C. Napoli, M. Woźniak, G. Susi, A spiking neural network-based long-term prediction system for biogas production (2020) Neural Networks, 129, pp. 271 - 279.

[11] Brandizzi N., Bianco V., Castro G., Russo S., Wajda A., Automatic RGB Inference Based on Facial Emotion Recognition (2021) CEUR Workshop Proceedings, 3092, pp. 66 - 74,

[12] G. Lo Sciuto, G. Capizzi, R. Shikler, C. Napoli, Organic solar cells defects classification by using a new feature extraction algorithm and an ebnn with an innovative pruning algorithm, International Journal of Intelligent Systems 36 (2021) 2443–2464.

[13] D. Połap, M. Woźniak, Meta-heuristic as manager in federated learning approaches for image processing purposes, Applied Soft Computing (2021) 107872.

[14] D. Połap, Analysis of skin marks through the use of intelligent things, IEEE Access 7 (2019) 149355–149363

[15] Caggiano, G., Napoli, C., Coretti, C. et al., Mold contamination in a controlled hospital environment: a 3-year surveillance in southern Italy. BMC Infect Dis 14, 595 (2014).

[16] Acciarito, S., Cristini, A., Di Nunzio, L., Khanal, G.M., Susi, G. An a VLSI driving circuit for memristor-based STDP (2016) 2016 12th Conference on Ph.D. Research in Microelectronics and Electronics, PRIME 2016, art. no. 7519503, .