ISSN: 2993-5377

Research Article

Open Access Journal of Applied Science and Technology

Fuzzy Numbers Unraveling the Intricacies of Neural Network Functionality

Vehbi Ramaj[1], Rame Elezaj[2] and Elvir Cajic[3*]

[1]*Busniess Faculty University Haxhi Zeka, Republic of Kosova*

[2]*Busniess Faculty University Haxhi Zeka, Republic of Kosova*

[3]*European University Kallos Tuzla, Bosnia and Herzegovina*

[*]Corresponding Author
Elvir Cajic, European University Kallos Tuzla, Bosnia and Herzegovina.

Submitted: 2023, Dec 15; **Accepted:** 2024, Jan 05; **Published:** 2024, Jan 17

Citation: Ramaj, V., Elezaj, R., Cajic, E. (2024). Fuzzy Numbers Unraveling the Intricacies of Neural Network Functionality. *OA J Applied Sci Technol, 2*(1), 01-07.

Abstract

This research delves into the synergy between fuzzy numbers and neural networks, presenting a novel perspective on interpreting neural network functionality. Fuzzy numbers offer a flexible framework to capture uncertainties and imprecisions, enriching the interpretability of neural network outputs. By integrating fuzzy number theory into the analysis, our study seeks to enhance the transparency and reliability of neural network models, contributing to a more nuanced understanding of their inner

Keywords: Fuzzy Numbers, Neural Networks, Interpretability, Uncertainty Modeling, Computational Intelligence

1. Introduction

Fuzzy logic refers to a mathematical theory based on the concept of vagueness and uncertainty. This theory is often used in combination with neural networks to obtain more accurate and reliable results. Fuzzy logic provides the ability to describe variables that are not clearly defined, which is often the case in real world problems. This paper will describe the application of fuzzy numbers in the interpretation of the work of neural networks with the use of Matlab software.

2. Definition of Fuzzy Number

A fuzzy number is a mathematical concept used in uncertainty theory. It is a number that has an indeterminate value that is between two limits. In other words, a fuzzy number is a number that has uncertainty in its value, which can be described using a membership function. A membership function is a mathematical tool used to describe uncertainty in values. It determines how much an element is connected to a group. In the case of fuzzy numbers, the membership function is used to describe how closely a value is related to a number. Fuzzy numbers are often used in various fields, such as finance, engineering, data science, etc. For example, in finance they are used for risk modeling and market price estimation. In engineering, they are used to model uncertainty in design and planning. In data science, they are used to analyze data that is not completely precise. There are various operations that can be applied to fuzzy numbers, such as addition, subtraction, multiplication, and division, which are performed using membership functions.

3. Membership Function

A membership function is a mathematical concept used in uncertainty theory and fuzzy logic. It describes how much an element is connected to a group, that is, how much an element "belongs" to a certain group. In the case of fuzzy numbers, the membership function describes how closely a value is related to a number.

The membership function is defined on the interval of values to which the elements belong, and its output is a value between 0 and 1, where 0 indicates that the element does not belong to the set, and 1 indicates that the element fully belongs to the set. For each element value, the membership function can be graphed.

For example, the membership function for the fuzzy number "height" can be as follows:

- Height: Low, the membership function would be close to 1 for values less than 160 cm, and closer to 0 for values greater than 180 cm.
- Height: Medium, the membership function would be close to 1 for values between 160 and 180 cm, and closer to 0 for values outside that interval.
- Height: Tall, the membership function would be close to 1 for values greater than 180 cm, and closer to 0 for values less than 160 cm.

A membership function can also be used to define sets that are not tightly defined, such as the sets "warm", "cold", "temperate" for the fuzzy number "temperature".

For example, the membership function for the fuzzy number "temperature" can be as follows:

- Temperature: cold, the membership function would be close

to 1 for values less than 10°C and closer to 0 for values greater than 20°C.
• Temperature: moderate, the membership function would be close to 1 for values between 10°C and 20°C, and closer to 0 for values outside that interval.
• Temperature: warm, the membership function would be close to 1 for values greater than 20°C and closer to 0 for values less than 10°C.

In this example, the membership function for the fuzzy number "temperature" can be used to make room air conditioning decisions, where setting the temperature between 10°C and 20°C is considered "moderate" and temperatures outside this interval are considered "cold" or "warm"

The membership function can be used in neural networks as an activation function, which determines how much each neuron in the network will be activated for a given input. Some of the common membership functions used in neural networks are sigmoid function, tangent hyperbolic function, ReLU function, etc..

For example, the sigmoid membership function is often used in neural networks and is defined as:

$$f(x) = 1 / (1 + e^{-x})$$

This function has an output between 0 and 1, and its curve has the shape of the letter "s". The sigmoid membership function is used in neural networks for classification and regression, and its main advantage is that it can take values of any size and fit them to an output that lies between 0 and 1.

In Matlab, you can use the sigmoid membership function as follows:
% defining the sigmoid membership function function y = sigmoid(x) y = 1./(1+exp(-x)); end
% applying the sigmoid membership function to the vector x x = [-5:0.1:5]; y = sigmoid(x);
% drawing graphs plot(x, y); title('Sigmoid membership function '); xlabel('x'); ylabel('y');
This code will define a sigmoid membership function, apply it to the vector x, and plot the graph of the function. It is important to note that the membership function can change depending on the specific application and needs of the neural network.

4. Application of Fuzzy Numbers

Fuzzy numbers are used in various applications where it is necessary to handle uncertain data and not just exact numbers. Some examples of applications of fuzzy numbers include:
1. Process control: Fuzzy numbers are used in automatic process control to optimize processes based on uncertain and variable parameters, such as temperature, humidity and flow rate.
2. Inventory Management: Fuzzy numbers are used in inventory management to better handle uncertain demand and variable lead times.
3. Finance: Fuzzy numbers are used in finance for risk management and investment optimization. For example, fuzzy numbers can be used to estimate the risk of investing in stocks, bonds or real estate.
4. Risk Management: Fuzzy numbers are used in risk management in various industries, including insurance, banking and security.
5. Quality management: Fuzzy numbers are used in quality management to manage uncertain variables, such as product quality and service quality.
6. Medicine: Fuzzy numbers are used in medicine for diagnosis and therapy, as well as for risk assessment in individuals.
7. Software Development: Fuzzy numbers are used in software development to handle uncertain requirements and variable conditions.

The application of fuzzy numbers in these and other areas can be performed through mathematical methods such as fuzzy logic, fuzzy inference and other tools.

5. Interesting Examples of Applications to Us Are in Medicine:

1. Diagnosis and therapy of diabetes: Fuzzy numbers can be used for diagnosis and therapy of diabetes. For example, fuzzy rules can be applied to blood sugar level data to diagnose diabetes and determine the optimal dose of insulin for a patient.

```
% Input data: blood sugar level
blood_sugar = 150;

% Fuzzy sets for blood sugar level
low = trapmf(blood_sugar, [0, 0, 70, 90]);
normal = trimf(blood_sugar, [80, 110, 140]);
high = trapmf(blood_sugar, [130, 150, 300, 300]);

% Fuzzy rules for the diagnosis of diabetes
rule1 = min(low, low);
rule2 = min(low, normal);
rule3 = min(normal, normal);
rule4 = min(normal, high);

% Output fuzzy sets for diabetes diagnosis
no_diabetes = max(rule1, rule2);
pre_diabetes = rule3;
diabetes = rule4;

% Graphic representation of fuzzy sets and output fuzzy sets
figure;
subplot(2,2,1);
plot(blood_sugar, low, 'b', [0 300], [0 0], 'k--');
title('Low Blood Sugar');
subplot(2,2,2);
plot(blood_sugar, normal, 'g');
title('Normal Blood Sugar');
subplot(2,2,3);
plot(blood_sugar, high, 'r', [0 300], [0 0], 'k--');
title('High Blood Sugar');
subplot(2,2,4);
plot(blood_sugar, no_diabetes, 'b', blood_sugar, pre_diabetes, 'g', blood_sugar, diabetes, 'r');
title('Diabetes Diagnosis');
```

2. Estimation of heart attack risk: Fuzzy numbers can be used to estimate the risk of heart attack in individuals. For example, fuzzy rules can be applied to data on age, blood pressure, cholesterol levels and other risk factors to estimate the overall risk of a heart attack.

```
% Input data: age, blood pressure, cholesterol level, physical activity and smoking
age = 55;
blood_pressure = 135;
cholesterol = 200;
activity_level = 3;
smoking = 1;

% Fuzzy sets for age
young = trapmf(age, [0 0 25 35]);
middle_aged = trimf(age, [25 45 65]);
old = trapmf(age, [55 70 100 100]);

% Fuzzy sets for blood pressure
low_blood_pressure = trapmf(blood_pressure, [0 0 90 110]);
normal_blood_pressure = trimf(blood_pressure, [90 120 150]);
high_blood_pressure = trapmf(blood_pressure, [130 160 300 300]);

% Fuzzy sets for cholesterol level
low_cholesterol = trapmf(cholesterol, [0 0 150 200]);
normal_cholesterol = trimf(cholesterol, [150 200 250]);
high_cholesterol = trapmf(cholesterol, [200 250 500 500]);

% Fuzzy sets for physical activity
sedentary = trapmf(activity_level, [0 0 1 2]);
moderately_active = trimf(activity_level, [1 3 5]);
active = trapmf(activity_level, [4 6 10 10]);

% Fuzzy smoking sets
non_smoker = trapmf(smoking, [0 0 0 1]);
smoker = trapmf(smoking, [1 1 2 2]);

% Fuzzy rules for heart attack risk assessment
rule1 = min(middle_aged, normal_blood_pressure);
rule2 = min(middle_aged, high_blood_pressure);
rule3 = min(middle_aged, high_cholesterol);
rule4 = min(middle_aged, moderately_active);
rule5 = min(old, low_blood_pressure);
rule6 = min(old, high_cholesterol);
rule7 = min(old, sedentary);
rule8 = min(old, smoker);

% Output fuzzy set for heart attack risk estimation
heart_attack_risk = max(rule1, max(rule2, max(rule3, max(rule4, max(rule5, max(rule6, max(rule7, rule8)))))));

% Graphic representation of fuzzy sets and output fuzzy set
figure;
subplot(2,3,1);
plot(age, young, 'b', age, middle_aged, 'g', age, old, 'r');
title('Age');
subplot(2,3,2);
plot(blood_pressure, low_blood_pressure, ');
```

3. Cancer Risk Assessment: Fuzzy numbers can be used to assess the risk of cancer in individuals. For example, fuzzy rules can be applied to data on age, sex, family medical history, lifestyle, and other risk factors to estimate the overall risk of developing cancer..

```
% Input data: age, gender, family history, lifestyle
age = 50;
gender = 'M';
family_history = 'Yes';
lifestyle = 'Unhealthy';

% Fuzzy sets for age
young = trapmf(age, [0 0 30 40]);
middle_aged = trimf(age, [30 50 70]);
old = trapmf(age, [60 80 100 100]);

% Fuzzy sets for gender
male = trapmf(gender, [0 0 1 1]);
female = trapmf(gender, [0 1 1 1]);

% Fuzzy sets for family history
no_history = trapmf(family_history, [0 0 0.5 0.75]);
yes_history = trapmf(family_history, [0.5 0.75 1 1]);

% Fuzzy sets for lifestyle
healthy = trapmf(lifestyle, [0 0 0.25 0.5]);
moderate = trimf(lifestyle, [0.25 0.5 0.75]);
unhealthy = trapmf(lifestyle, [0.5 0.75 1 1]);

% Fuzzy rules for assessing the risk of developing cancer
rule1 = min(young, female);
rule2 = min(middle_aged, male);
rule3 = min(old, male);
rule4 = min(middle_aged, yes_history);
rule5 = min(unhealthy, middle_aged);
rule6 = min(high_pressure, middle_aged);

% Output fuzzy set for cancer risk assessment
cancer_risk = max(rule1, max(rule2, max(rule3, max(rule4, max(rule5, rule6)))));

% Graphic representation of fuzzy sets and output fuzzy set
figure;
subplot(2,2,1);
plot(age, young, 'b', age, middle_aged, 'g', age, old, 'r');
title('Age');
subplot(2,2,2);
plot(gender, male, 'b', gender, female, 'r');
title('Gender');
subplot(2,2,3);
plot(family_history, no_history, 'b', family_history, yes_history, 'r');
title('Family History');
subplot(2,2,4);
plot(lifestyle, healthy, 'b', lifestyle, moderate, 'g', lifestyle, unhealthy, 'r');
```

```
title('Lifestyle');
figure;
plot(cancer_risk);
title('Cancer Risk');
```

6. Operations With Fuzzy Numbers

Fuzzy Number Operations: There are basic mathematical operations that can be applied to fuzzy numbers, including addition, subtraction, multiplication, and division. These operations can be performed using membership functions of fuzzy sets and standard mathematical operators.

For example, if two fuzzy numbers A and B are added, the membership functions of the sets A and B are added to obtain a new membership function for the fuzzy number A + B.

```
% Fuzzy numbers A and B
A = [0.4 0.7 0.9];
B = [0.2 0.5 0.8];

% The addition of fuzzy numbers A and B
sum_A_B = A + B;

% Graphic representation of fuzzy numbers A, B and A + B
figure;
subplot(1,3,1);
plot(A, 'b');
title('A');
subplot(1,3,2);
plot(B, 'r');
title('B');
subplot(1,3,3);
plot(sum_A_B, 'g');
title('A + B');

% Subtraction of fuzzy numbers A and B
diff_A_B = A - B;

% Graphic representation of fuzzy numbers A, B and A - B
figure;
subplot(1,3,1);
plot(A, 'b');
title('A');
subplot(1,3,2);
plot(B, 'r');
title('B');
subplot(1,3,3);
plot(diff_A_B, 'g');
title('A - B');

% Multiplication of fuzzy numbers A and B
mult_A_B = A * B;

% Graphic representation of fuzzy numbers A, B and A * B
figure;
subplot(1,3,1);
plot(A, 'b');
title('A');
subplot(1,3,2);
plot(B, 'r');
title('B');
subplot(1,3,3);
plot(mult_A_B, 'g');
title('A * B');

% Division of fuzzy numbers A and B
div_A_B = A ./ B;

% Graphic representation of fuzzy numbers A, B and A / B
figure;
subplot(1,3,1);
plot(A, 'b');
title('A');
subplot(1,3,2);
plot(B, 'r');
title('B');
subplot(1,3,3);
plot(div_A_B, 'g');
title('A / B');
```

The membership function of the fuzzy numbers 1, 3, 5, 7 and 9 depends on the selected membership function. Here I will show an example of defining fuzzy numbers using the triangular membership function in MATLAB:

```
% Defining fuzzy numbers 1, 3, 5, 7 and 9 using triangular membership functions
x = 0:0.1:10;
mu1 = trimf(x, [0 0 2]);
mu3 = trimf(x, [2 4 6]);
mu5 = trimf(x, [4 6 8]);
mu7 = trimf(x, [6 8 10]);
mu9 = trimf(x, [8 10 10]);

% Graphic representation of fuzzy numbers
plot(x, mu1, 'b', x, mu3, 'r', x, mu5, 'g', x, mu7, 'm', x, mu9, 'k');
legend('1', '3', '5', '7', '9');
xlabel('x');
ylabel('\mu(x)');
```

This script will create a membership function graph for the fuzzy numbers 1, 3, 5, 7, and 9 using triangular membership functions. If you want to use other membership functions, you need to change the parameters of the **trimf** function accordingly.

Fuzzy numbers can be used in neural networks to improve prediction accuracy. Using fuzzy logic, a neural network can take into account uncertainties and ambiguities in the data and provide probabilistic outputs.

One way to use fuzzy numbers in neural networks is to use a membership function to calculate probabilities. For example, if the input is a person's height, the membership function can give the probability that the person is tall, medium height, or short.

Another way to use fuzzy numbers in neural networks is to use fuzzy number operations to improve precision. For example, if a

fuzzy number is used to calculate a value, an operation with that number can give a more precise value than using only a single number.

Code

```
% Loading data for training a neural network
load('training_set.mat');

% Input and output normalization
training_input = normalize(training_set(:, 1:end-1));
training_output = normalize(training_set(:, end));

% Defining a membership function for input data
height_fuzzy = newfis('height_fuzzy');
height_fuzzy = addvar(height_fuzzy, 'input', 'height', [0 200]);
height_fuzzy = addmf(height_fuzzy, 'input', 1, 'short', 'gaussmf', [15 0]);
height_fuzzy = addmf(height_fuzzy, 'input', 1, 'average', 'gaussmf', [15 100]);
height_fuzzy = addmf(height_fuzzy, 'input', 1, 'tall', 'gaussmf', [15 200]);

% Neural network training
net = newff(training_input', training_output', [5 1], {'tansig' 'purelin'});
net = train(net, training_input', training_output');

% Testing neural networks with fuzzy input data
test_input = [fuzzify(165, height_fuzzy) fuzzify(70, weight_fuzzy)];
test_output = sim(net, test_input');
test_output = defuzz(test_output, height_fuzzy);

% Printing the results
fprintf('Predicted height: %.2f\n', test_output);
```

In this example, a membership function for a person's height is used, which has three fuzzy sets: "short", "average" and "tall". After training the neural network, it was tested with fuzzy input data for a person's height and weight. By using the fuzzify function, the input data is transformed into fuzzy numbers according to the membership function. The final output of the neural network.

7.Use of Fuzzy Numbers in Neuronal Networks

One way to use fuzzy numbers in neural networks is to use fuzzy logic for decision making. Another way is to use operations with fuzzy numbers to improve classification accuracy.

An example of using fuzzy numbers to improve classification accuracy would be as follows. Suppose we want to classify a type of fruit (apple, banana, orange) based on their characteristics (color, shape, size). A normal neural network could use a classifier that will give one classification for each fruit (apple, banana or orange).

However, by using fuzzy numbers as input variables to the neural network, we can improve the classification accuracy. For example, instead of using only the color of the fruit as an input variable, we can use a fuzzy number for the color, which will allow us to take into account the shades of the color of the fruit..

The MATLAB code for this example would look something like this:

```
% Loading fruit datasets (color, shape, size)
data = load('fruit_data.mat');

% Defining a fuzzy set for a color
color = [fuzzy(0, 0, 0.25, 0.5), fuzzy(0.25, 0.5, 0.75), fuzzy(0.5, 0.75, 1), fuzzy(0.75, 1, 1, 1)];

% Defining a neural network
net = feedforwardnet([10, 5]);

% Configuring a neural network to use fuzzy inputs
net.inputs{1}.processFcns = {'mapminmax', 'fuzzyfication'};
net.inputs{1}.processParams{2} = color;

% Neural network training
net = train(net, data.inputs, data.targets);

% Testing the neural network on new data
new_data = [0.6, 0.8, 0.4];
output = net(new_data);
```

In this code, we defined a fuzzy set for the color of the fruit and configured the neural network to use fuzzy inputs. We then trained the neural network and tested it on new data. By using fuzzy numbers for the inputs of the neural network, we improve the precision of the classification, because we take into account more shades of fruit color.

Example

1. Forecasting real estate prices using a neural network with fuzzy logic:

Real estate price forecasting using a neural network with fuzzy logic can be a very effective way to predict the market value of real estate. Fuzzy logic is used to deal with the indeterminacy in the data inherent in this problem, such as subjective opinions about real estate value and unstructured data.

A neural network is used to learn from real estate data, such as square footage, location, number of rooms, year of construction, and the like. This network could be trained on a real estate dataset, taking into account not only the facts about the property, but also people's subjective opinions about the value of the property.

Fuzzy logic is used to process these subjective opinions, such as people's perceptions of real estate value, which cannot be expressed in exact numbers. Instead, fuzzy logic is used to model uncertainty and imprecision in this data, which allows the neural network to learn to adapt to such variations in the data.

Ultimately, a neural network with fuzzy logic can predict the market value of a property based on input property data, taking

into account people's subjective assessments of the property's value. This technique can be very effective in predicting real estate prices, especially in situations where data is unclear or missing.

```
% Loading data for neural network training
trainData = load('trainData.mat');
trainInputs = trainData.inputs; % input data
trainTargets = trainData.targets; % expected output data

% Defining a neural network with fuzzy logic
numInputs = size(trainInputs, 2); % number of input features
numOutputs = size(trainTargets, 2); % number of output features
numMFs = 3; % number of membership functions in fuzzy logic

inputMFs = zeros(numInputs, numMFs); % membership functions of the input data
outputMFs = zeros(numOutputs, numMFs); % membership functions of output data

% Defining membership functions for input data
for i = 1:numInputs
    inputMFs(i, :) = trimf(trainInputs(:, i), [min(trainInputs(:, i)) mean(trainInputs(:, i)) max(trainInputs(:, i))]);
end

% Defining output membership functions
for i = 1:numOutputs
    outputMFs(i, :) = trimf(trainTargets(:, i), [min(trainTargets(:, i)) mean(trainTargets(:, i)) max(trainTargets(:, i))]);
end

% Defining a neural network
fis = genfis1(trainInputs, trainTargets, 'gaussmf', numMFs);

% Neural network training
options = anfisOptions('InitialFIS', fis, 'EpochNumber', 100, 'ValidationData', [trainInputs, trainTargets]);
model = anfis([trainInputs, trainTargets], options);

% Testing the model on new data
testData = load('testData.mat');
testInputs = testData.inputs;
testTargets = testData.targets;
testOutputs = evalfis(testInputs, model);

% Calculating model error metrics
rmse = sqrt(mean((testTargets - testOutputs).^2));
fprintf('Root Mean Square Error (RMSE): %.2f\n', rmse);
```

This code is based on using the fuzzy logic processing tool, **genfis1** function and **anfisOptions** to generate and train the model. The trimf function is also used to define membership functions for triangular functions. These functions can be customized depending on your needs and the requirements of the real estate price forecasting problem

8. The Following Code Shows the Comparison Between Actual Values and Predicted

```
% Loading data for neural network training
trainData = load('trainData.mat');
trainInputs = trainData.inputs; % input data
trainTargets = trainData.targets; % expected output data

% Defining a neural network with fuzzy logic
numInputs = size(trainInputs, 2); % number of input features
numOutputs = size(trainTargets, 2); % number of output features
numMFs = 3; % number of membership functions in fuzzy logic

inputMFs = zeros(numInputs, numMFs); % membership functions of the input data
outputMFs = zeros(numOutputs, numMFs); % membership functions of output data

% Defining membership functions for input data
for i = 1:numInputs
    inputMFs(i, :) = trimf(trainInputs(:, i), [min(trainInputs(:, i)) mean(trainInputs(:, i)) max(trainInputs(:, i))]);
end

% Defining output membership functions
for i = 1:numOutputs
    outputMFs(i, :) = trimf(trainTargets(:, i), [min(trainTargets(:, i)) mean(trainTargets(:, i)) max(trainTargets(:, i))]);
end

% Defining a neural network
fis = genfis1(trainInputs, trainTargets, 'gaussmf', numMFs);

% Neural network training
options = anfisOptions('InitialFIS', fis, 'EpochNumber', 100, 'ValidationData', [trainInputs, trainTargets]);
model = anfis([trainInputs, trainTargets], options);

% Testing the model on new data
testData = load('testData.mat');
testInputs = testData.inputs;
testTargets = testData.targets;
testOutputs = evalfis(testInputs, model);

% Calculating model error metrics
rmse = sqrt(mean((testTargets - testOutputs).^2));
fprintf('Root Mean Square Error (RMSE): %.2f\n', rmse);

% Graphic representation of the comparison of actual and predicted values
figure;
plot(testTargets, 'r');
hold on;
plot(testOutputs, 'b');
legend('Stvarne vrijednosti', 'Predicted values ');
xlabel('Number of examples ');
ylabel('The price of real estate ');
title('Comparison of actual and predicted values ');
```

9. Conclusion of The Work

In the conclusion of the paper on the application of fuzzy numbers in the interpretation of the work of neural networks with the use of Matlab software, several key findings and conclusions can be highlighted.

First, the application of fuzzy numbers in the interpretation of the work of neural networks can be useful for assessing uncertainty in data and for understanding the role of individual variables in the decision-making process. Fuzzy logic makes it possible to evaluate variables that are not completely defined, which is often the case in real situations.

Second, Matlab software can be a useful tool for implementing fuzzy logic and neural networks. Matlab offers many functions and tools that allow users to easily model and analyze complex systems.

Third, this paper presents examples of the application of fuzzy numbers in the interpretation of the work of neural networks for solving problems from different areas, including production process management and real estate price forecasting. These examples show that fuzzy logic and neural networks can be successfully applied in different situations.

Finally, this paper provides useful guidelines and examples for the application of fuzzy logic and neural networks in real-world situations. These tools can be useful in many fields, including engineering, science, economics, and others, where we encounter uncertainties and complex systems [1-24].

References

1. Ross, T. J. (2009). Fuzzy logic with engineering applications. John Wiley & Sons.
2. Jang, J. S. R., Sun, C. T., Mizutani, E., & Ho, Y. C. (1998). Neuro-fuzzy and soft computing-a computational approach to learning and machine intelligence. PROCEEDINGS-IEEE, 86(3), 600-603.
3. Haykin, S. (1998). Neural networks: a comprehensive foundation. Prentice Hall PTR.
4. Matlab. (2023). The MathWorks, Inc. Retrieved from https://www.mathworks.com/products/matlab.html
5. Negoita, M. G. (2012). Fuzzy logic-based control systems: fuzzy logic controller design for uncertain nonlinear systems.
6. Chen, H., & Liu, Y. (2004). A fuzzy neural network approach to real estate price forecasting. Appl. Soft Comput, 24, 649-656.
7. Bezdek, J. C. (2013). Pattern recognition with fuzzy objective function algorithms. Springer Science & Business Media.
8. Dubois, D. J. (1980). Fuzzy sets and systems: theory and applications (Vol. 144). Academic press.
9. Mamdani, E. H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. International journal of man-machine studies, 7(1), 1-13.
10. Kubat, M. (1995). Neural networks and fuzzy systems: A dynamical systems approach to machine intelligence by Bart Kosko, Prentice Hall, Englewood Cliffs, NJ, 1992, pp 449,£ 24.96, ISBN 0-13-612334. The Knowledge Engineering Review, 10(2), 219-220.
11. Klir, G., & Yuan, B. (1995). Fuzzy sets and fuzzy logic (Vol. 4, pp. 1-12). New Jersey: Prentice hall.
12. Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. Pattern recognition, 26(9), 1277-1294.
13. Pedrycz, W. (1993). Fuzzy sets in engineering design. John Wiley & Sons.
14. Ripley, B. D. (2007). Pattern recognition and neural networks. Cambridge university press.
15. Bezdek, J. C., & Hathaway, R. J. (2002). Exponential radial basis functions and fuzzy clustering. Fuzzy Sets and Systems, 128(1), 123-140.
16. Beal, M. T., Hagan, H. B., & Demuth, M. (1996). Neural Network Design, PWS Pub. Co., Boston.
17. Lee, C. C. (1990). Fuzzy logic in control systems: fuzzy logic controller. I. IEEE Transactions on systems, man, and cybernetics, 20(2), 404-418.
18. Negoita, M. G., & Negoita, C. V. (2017). Fuzzy systems engineering: theory and practice. Springer.
19. Tadeusiewicz, R. (1993). Fuzzy control: fundamentals, stability and design of fuzzy controllers. Springer.
20. Wang, L. X. (1994). Adaptive fuzzy systems and control: design and stability analysis. Prentice-Hall, Inc..
21. E.Č D. Galić,Z.Stojanović (2024). Application of Neural Networks and Machine Learning in Image Recognition Tehnicki vjesnik - Technical Gazette 1 (31), 316-323
22. Čajić, E., Stijanović, Z., Galić, D.(2023). Investigation of delay and reliability in wireless sensor networks using the Gradient Descent algorithm IEE Telfor Belgrade
23. Čajić, E., Ibrišimović, I., Šehanović, A., Bajrić, D., Ščekić, J. (2023). Fuzzy Logic and Neural Networks for Disease Detection and Simulation in Matlab
24. Galić, R., Čajić, E. (2024). Optimization and Component Linking Through Dynamic Tree Identification (DSI), https://doi.org/10.21203/rs.3.rs-3601218/v1