Neural Network

Worlasi Kofi Dzam

05/2/2025

Fuzzy and Neural Network Controllers for Non-Linear Surface Approximation

Abstract

Fuzzy systems are machine learning models that handle varying degrees of truth often found in uncertain or imprecise data (non-linear real-world scenarios). Much like Neural Networks, which are inspired by the operations of biological neurons, both systems are very powerful tools in making key decisions and inferences. In this project we first design a fuzzy controller based on the given control surface and training data. Next, we train a neural network to approximate the same surface and then compare the performance and modeling capabilities of both models using Matrix Laboratory (MATLAB) version R2024b.

Introduction

Neural networks are computational systems that act like biological neurons. They allow machines the ability to learn from data and then apply the knowledge to new situations. Fuzzy systems are another form of machine learning used to model and control complex systems that have certain “gray areas” where traditional Yes or No (Binary logic) is insufficient. They enable modeling in the presence of uncertainty or partial truth by assigning degrees of membership to variables such as “low or high”.

Part 1

To design the Fuzzy system, we must go through the systematic process of;

1. Fuzzification: converting raw numerical input values into fuzzy numbers (membership values). These values express the degree to which an input belongs to a stated fuzzy set.
2. Fuzzy table or rules: Next we set our fuzzy rules or fuzzy table. These are input combinations and rules that define system behavior based on fuzzy logic. For example, if temperature is high AND vapor is low, then microwave state is off or if temperature is low and vapor is low, the microwave state is on.
3. We then perform a Minimum (for AND) and Maximum (for OR) computation of our fuzzy table or rules to infer the outputs. For example, in the Mamdani system of fuzzy

controllers, say if temperature =0.9(hot) and vapor = 0.1(low) then state of microwave= min (0.9,0.1) = 0.1 which means the microwave should stop as defined by our fuzzy rule or fuzzy table.

4. Defuzzification: defuzzification is the process of converting fuzzy outputs into a single crisp value. In Mamdani type controllers this is done by computing the weighted average of the max values from the rule table, In the Takagi-Sugeno-Kang fuzzy type controllers we do same but using all the associated output values instead of the fuzzy sets from the rule table.

Assessing the Data

Based on the given error surface and the data we observe that both input 1, input 2 and the output have values between 0 to 4. this is important to us to design the right number of member functions which tell us how much of an input belongs to a certain class or level. There are 3 options of Triangular, Trapezoidal and Gaussian Membership Functions. However, based on my judgement from the data set a triangular membership function would be of best suit as it generally has a simpler and faster computation model

Our input and output variables are set up to match every possible combination to obtain the rule table. I prefer to go with the Mamdani type fuzzy system with triangular Membership functions, since my inputs have well defined peaks (0,1,2,3,4) trapezoidal would have been better for input values that have continuous values.

For the required outputs the values were rounded up so they can be organized into categories as well for simplification. While this reduces precision slightly, it improves interpretability and rule base construction.

Output Value Range	**Quantized Level**	**Fuzzy Output Label**
0 – 0.60	0.5	Very Low
0.61 - 1.4	1.5	Low
1.41 - 2.9	2.5	Normal
> 2.9	3.5	High

Output Value	**Quantized level**	**Category**
0.055648	0.5	Very Low

0.18476	0.5	Very Low
0.33665	0.5	Very Low
0.33665	0.5	Very Low
0.18476	0.5	Very Low
0.15902	0.5	Very Low
0.52798	1.5	Low
0.96203	1.5	Low
0.96203	1.5	Low
0.52798	1.5	Low
0.33665	0.5	Very Low
1.117	1.5	Low
2.0366	2.5	Normal
2.0366	2.5	Normal
1.117	1.5	Low
0.52798	1.5	Low
1.7529	1.5	Low
3.1941	3.5	High
3.1941	3.5	High
1.7529	1.5	Low
0.61342	1.5	Low
2.0366	2.5	Normal
3.711	3.5	High
3.711	3.5	High
2.0366	2.5	Normal

The inputs are discrete making it easier to directly label them, the labelling is as follows:

0 = Very Low

1 =Low

2 = Medium

3 = High

4 = Very High

The fuzzy rule table is then derived from the 25 training data points given to us:

Fuzzy Rule Table

Input1/Input2	Very Low	Low	Medium	High	Very High

Very Low	Very Low	Very Low	Very Low	Very Low	Very Low
Low	Very Low	Very Low	Low	Low	Very Low
Medium	Very Low	Low	Normal	Normal	Low
High	Very Low	Normal	High	High	Normal
Very High	Low	Normal	High	High	Normal

Input1/Input2	0	1	2	3	4
0	1	1	1	1	1
1	1	1	2	2	1
2	1	2	3	3	2
3	1	3	4	4	3
4	2	3	4	4	3

A Matrix Laboratory (MATLAB) code was run to implement it, and the 3D approximation is result is as shown below:

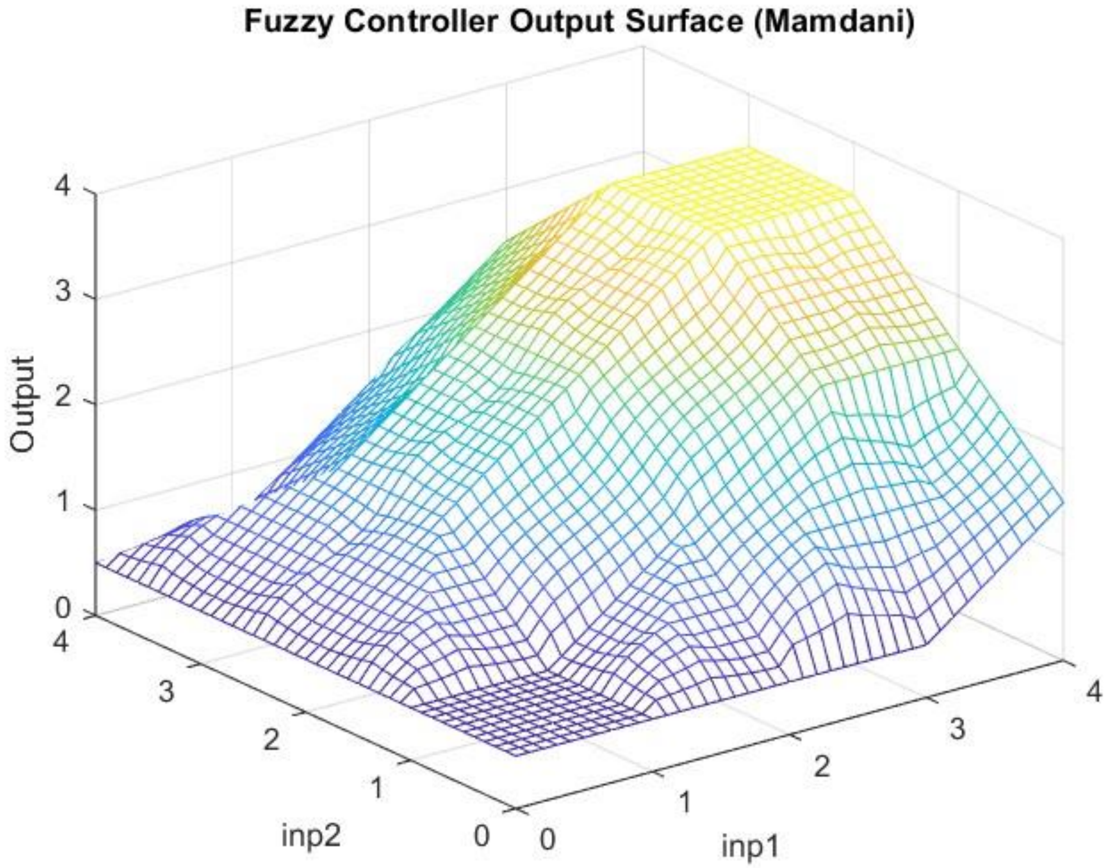

Figure 1: Fuzzy rule surface generated using Mamdani system.

As seen from the plot above, the fuzzy controller has a relatively good approximation of the error surface, the blocky nature of the surface is due to sharp transitions on the error surface as result quantization errors introduced to simplify the system. It had an MSE of 0.96.

Part 2

Neural Network Training

The goal of this task is to approximate the same control surface using a neural network. We implemented a Fully Connected Cascade (FCC) neural network using MATLAB's Deep Learning Toolbox. The architecture included 6 hidden neurons, 1 output neuron and a bias term, making a total of 8 neurons. The network uses the hyperbolic tangent (tanh) activation function to approximate the non-linear mapping between the two inputs and the output. Training was implemented using the Levenberg-Marquardt algorithm with a random data split of 80% for training and 20% for testing to ensure proper fitting. Finally, as a performance metric, the loss was computed using mean squared error (MSE).

A fully Connected Cascade network (FCC) is an extremely powerful network. Its architecture is unique in that not only does each layer receive input from the previous layer but also from every other earlier layer including the input layer. Every neuron's output is used multiple times as input to other neurons as you proceed deeper into its layers. This improves learning efficiency over models like the standard feedforward network even with fewer neurons making it suitable for such a function approximation task.

When implemented in MATLAB the resulting 3D surface is shown below:

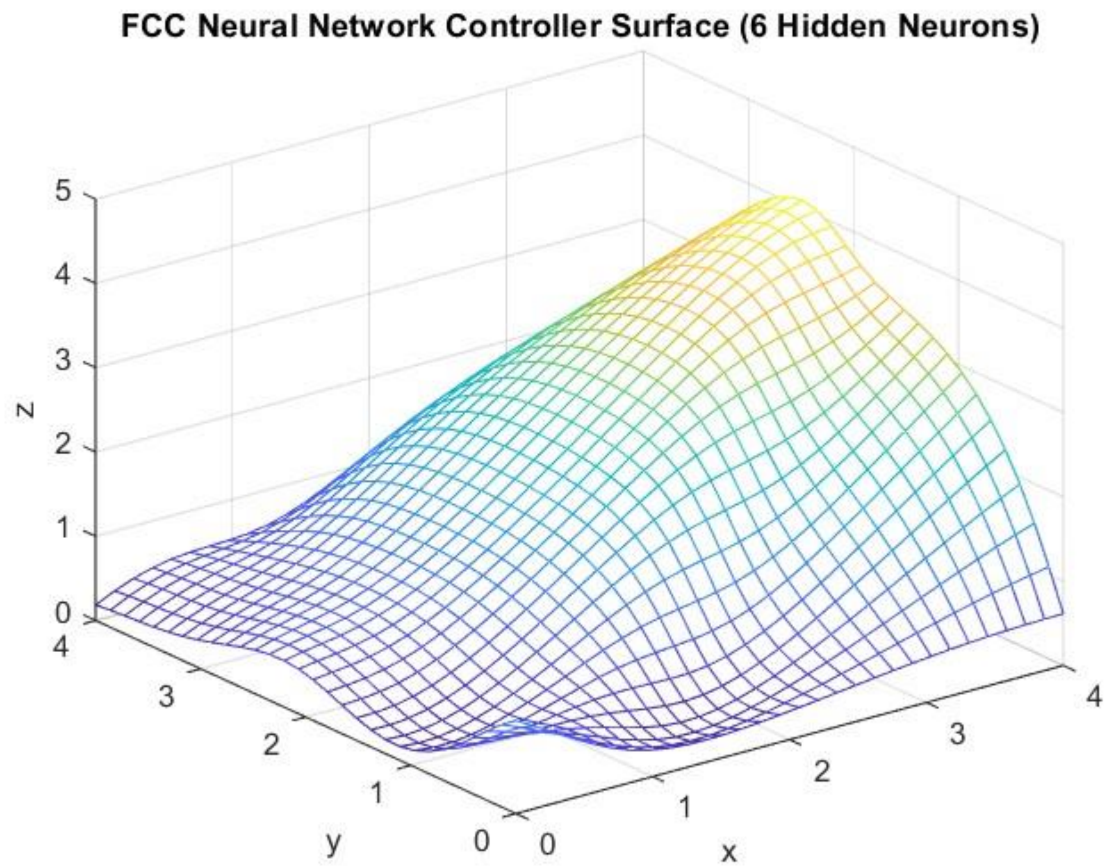

Figure 2: Neural Network Approximation of the Control Surface

As seen the FCC network has been able to almost perfectly approximate the given error surface, one much “smoother” than that of the Fuzzy system. With its unique architecture and design each neuron captures the necessary levels of complexities via the training data and implements a more accurate model. Performance-wise it had an MSE of 0.006425.

Conclusion

Both the Fuzzy Control system and the Neural network system show how powerful machine learning and artificial intelligence can be as ideal decision-making tools in our ever complex and dynamic world. Neural networks have the major advantage in terms of efficiency and performance as shown by our results proving why they are more sought after for most tasks in real world applications.

References:

[1] Gong, X. (2025). *Neural Network ELEC 6240 Lecture Notes*, Auburn University.

[2] MathWorks Documentation. Neural Network Toolbox. https://www.mathworks.com/products/neural-network.html

[3] Understanding Neural Networks - YouTube Tutorial. https://www.youtube.com/watch?v=__0nZuG4sTw