Applied Computational Intelligence

2021/2022

Prediction using NN and Fuzzy Systems

Labs 7-8: Project 1

Name: Alice Rosa N.º 90007
Name: Francisco Galante N.º 90073

Group Nº9

Teachers: João Paulo Carvalho, António Paiva Lapas de Gusmão

1 Introduction

This project aims to predict when will an IoT gateway device fail based on its processor load and the received requests that might requite data processing and can be computationally demanding. In order to do this, a MLP Neural Network and a Fuzzy Rule Based Inference System were implemented, taking into account some insights provided by two experts.

2 MLP - Brute Force Approach

In a first approach, we implement a Multi-layer Perceptron classifier (MLP) with 2 inputs, the normalized number of requests received by the device during the 20 min period and the processor average load during that period, and 1 output that is a binary variable with the value 0 when the device crashes and 1 to indicate the normal operation.

The first step is to read the data from the file given and then divide it into training and validation sets (70% and 30%, respectively). The training dataset is used to train candidate MLP models, the validation dataset is used to compare their performances and decide which one to take.

The hyperparameters used and optimized in this project are: the number of hidden layers, activation function, solver and learning rate. In the table 1 is presented the final values chosen for the hyperparameters and the results obtained for the classification metrics with the validation set: Precision, Recall, F-measure and confusion matrix.

Table 1: Hyperparameters chosen for the MLP model

Number of hidden layers	Activation function	Solver	Learning rate
200	relu	adam	0.002

Table 2: Precision, Recall and F-measure obtained with Brute Force Approach

F1 Score	Precision	Recall
0.606	0.588	0.625

Table 3: Confusion matrix obtained for Brute Force Approach

		Predicted		
		Negative	Positive	Total
Actual	Negative	577	7	584
	Positive	6	10	16
	Total	583	17	600

From table 2 and 3 we can already conclude that, even after the optimization of the hyperparameters, the results obtained for the metrics are low.

Throughout this work, our goal is to maximize the Recall metric, i.e, to maximize the number of True Positives (detect correctly a crash) and minimize the number of

False Negatives (does not detect when a crash occurs), since we think these are the most important cases to consider.

To reach this goal, we need to study better our data and make some changes to our inputs.

3 MLP - Expert 1

This expert in intelligent systems found that the dataset is highly imbalanced because of the lack of positive instances and his advice is to use some previous occurrences of the number of requests as additional inputs, together with the processor load input.

Firstly, the dataset is modified in order to add more features corresponding to previous instances. In this case, the number of previous instances that provided the best results is 3, so these were considered to use as inputs. This new dataset is now splitted into training set and validation set, being the percentages of each set the same as before. The hyperparameters used are presented in the Table 1.

Then, the training set is balanced by duplicating the minority class, which is the number of positive instances that indicate a crash operation. At this point, the model with new features added as inputs and a more balanced training set is trained. The results obtained for the classification metrics are presented in tables 4 and 5.

Table 4: Precision, Recall and F-measure obtained with Expert's 1 advice

F1 Score	Precision	Recall
0.9696	0.9411	1.0000

Table 5: Confusion matrix obtained with Expert's 1 advice

		Predicted		
		Negative	Positive	Total
Actual	Negative	582	1	583
	Positive	0	16	16
	Total	582	17	599

By analysing the classification metrics, it is possible to notice that the results obtained with Expert's 1 advice are very good, as the Recall metric is equal to 1, which means that the model never predicts a normal operation when it is, in fact, a gateway crash. Also, the model only predicts once a gateway crash instead of a normal operation. This proves that, considering Expert's 1 insights, the implemented model is good.

4 MLP - Expert 2

This expert suggests that we try to look for an abnormal number of requests in the periods before the processor load gets noticeably high.

For this effect, a new feature that is the weighted average of the number of request in the N previous instances is created. For this, it is also created a weight vector that gives more importance to the latest instances than the old ones.

After altering the inputs, the dataset is divided into training and validation set, in the same propositions as previously done, and then balanced in the same way as before for expert 1.

The tunning of this model is done through two hyperparameters: number of previous instances of the number of requests to consider in the calculation of the weighted average (N request) and the weight vector used in these calculation.

After some evaluations done through the validation set, the final values chosen to these hyperparameters are presented in the table 6. The results obtained for the classification metrics are presented in tables 7 and 8.

Table 6: Hyperparameters chosen for the model with Expert's 2 advice

N request	Weight vector
4	[0.1 0.4 0.7 1]

Table 7: Precision, Recall and F-measure obtained with Expert's 2 advice

F1 Score	Precision	Recall
0.9677	1.0000	0.9375

Table 8: Confusion matrix obtained with Expert's 2 advice

		Predicted		
		Negative	Positive	Total
Actual	Negative	583	0	583
	Positive	1	15	16
	Total	582	17	599

From these results, we can conclude that the main goal (Recall=1) was almost obtained, but not fully. Probably, in that specific case the weights had to be increased so the crash would be detected. However, the precision obtained is 1, so we consider this a general good model to maintain.

5 Fuzzy Rule Based Expert System

This is the last method used to predict the crashes and it is also based on the knowledge provided by the Expert 2.

Firstly, the processor load and the average number of requests feature are chosen to be the inputs of the Fuzzy System as antecedents. Also, it is defined to be three linguistic terms for each input, being these *Low*, *Medium* and *High*. The consequent is the gateway operation, in which the linguistic terms are *Normal Operation* and *Crash*.

Then, membership functions are assigned to each antedecent and consequent, in order to determine the fuzzy sets. In this case, triangular membership functions are used.

The fuzzy sets can be visualized in Figures 1 and 2.

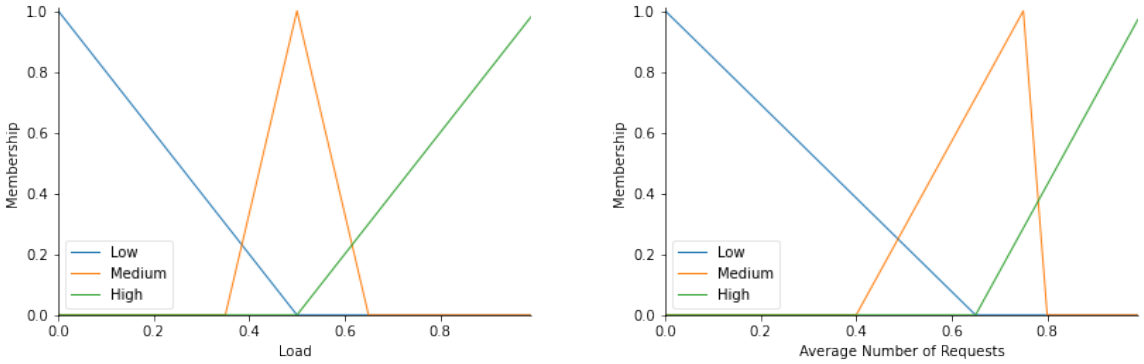

Figure 1: Inputs of the Fuzzy System

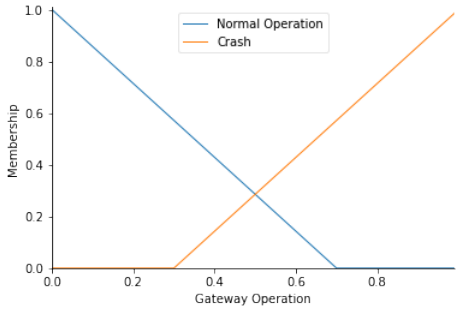

Figure 2: Output of the Fuzzy System

As there are 2 inputs and 3 linguistic terms, a total of $2^3 = 9$ rules to predict if the gateway operation is a crash are implemented.

Throughout the validation process of our fuzzy system, it was tested various intervals for the membership functions of both inputs and output. After checking our data, this intervals were accordingly changed in order to obtain better results in the validation sets. Some of the rules were also modified.

After passing values to the inputs of the control system with all the defined rules, the fuzzy system is simulated and the results obtained are the presented in tables 9 and 10.

Table 9: Precision, Recall and F-measure obtained from Fuzzy System with Expert's 2 advice

F1 Score	Precision	Recall
0.9485	0.9020	1.0000

Table 10: Confusion matrix obtained from Fuzzy System with Expert's 2 advice

		Predicted		
		Negative	Positive	Total
Actual	Negative	1944	5	1949
	Positive	0	46	46
	Total	1944	51	1995

From table 9, it is verifiable that the main goal (Recall=1) is reached. However, there were 5 cases were the system predicted a crash when there wasn't one. But we consider that the overall model has a good functioning.

6 Generalization

The new data obtained while the models were being developed is now used as a test set to evaluate how well our models perform with this new dataset that is not known from before.

The results obtained for each model (refereed in chapters 3, 4 and 5) are presented in the table 11, 12 and 13.

Table 11: Precision, Recall and F-measure obtained from the different models for the test dataset

Model	F1 Score	Precision	Recall
MLP - Expert 1	0.9411	0.8889	1.0000
MLP - Expert 2	0.9411	0.8889	1.0000
Fuzzy	0.9411	0.8889	1.0000

Table 12: Confusion matrix obtained with MLP - Expert's 1 advice for the test dataset

		Predicted		
		Negative	Positive	Total
Actual	Negative	187	1	188
	Positive	0	8	8
	Total	187	9	196

Table 13: Confusion matrix obtained with MLP - Expert's 2 advice and Fuzzy System for the test dataset

		Predicted		
		Negative	Positive	Total
Actual	Negative	186	1	187
	Positive	0	8	8
	Total	186	9	195

We conclude that, for this test set, all models developed present the same performance. The final goal was obtained for all of them, the Recall achieve is 1 for all, but an improvement in the precision metric could be done for all of the models. Concluding, the models generated adapt well to new data.

If we had to chose one NN to implement in a real case, we would probably go for the second model of the MLP (with expert's 2 advice) because it performs as well as the first one but with less features and is easy to implement. The fuzzy system is also a good choice to implement in real life. If the membership function is tunned even more (for example, add more linguistic terms) and the rules of the system, it is probable to obtain even better results to this system. However, the difficulty of implementation also increases.