Computational Intelligence for the IoT

Project 1.2

Classification using

Fuzzy Systems and NN

Professor:

João P. Carvalho

Students (G2) :

Aurora Nora | 93573

João Lopes | 93584

In this report we will explain how we developed a model using a Fuzzy System with the purpose of knowing if the number of students inside a lab is above the limit (2 students), based on data received from several sensors inside that specific room.

Initially, we analyzed the given dataset and the problem that we would have to solve in the development of the Fuzzy System. We also started thinking about which features would make sense to use this time, as we can't have so many inputs in fuzzy systems because of the complexity.

Features

We thought about using several combinations of features, like the light sensors (S*i*Light), CO2 variation, the time of day (day and night), the weather (sunny, cloudy) and many more possibilities. After some testing we ended up using only the light sensors and the CO2 sensor, in order to know when a room is empty. This are some of the features we tried to use:

Daytime: Separates the day in two parts: "day" and "night" by using the "Time" column.

- Day: 7am-7pm.
- Night: 7pm-7am.
- This column would be used to set higher/lower limits on the sum of lights in order to estimate how many lights were on.

Weather: It's composed by "S3Light" and "CO2".

- Sunny: SL3Light > 170 & CO2 < 400
- Cloudy – SL3Light < 170 & CO2 < 400
- This column would be used to set higher/lower limits on the sum of lights in order to estimate how many lights were on.

Lights Sum: Sum of the light sensors ("S1Light", "S2Light", "S3Light").

- Lab overcrowded: "night" & "sum_lights" > 700
- Lab undercrowded: "night" & "sum_lights" < 701
- Lab overcrowded: "day" & "sunny" & "sum_lights" > 1000
- Lab overcrowded: "day" & "sunny" & "sum_lights" < 1001
- Lab overcrowded: "day" & "cloudy" & "sum_lights" > 700
- Lab overcrowded: "day" & "cloudy" & "sum_lights" < 701

Lights Total: Amount of lights turned on using the light sensors ("S1Light", "S2Light", "S3Light").

- S1Light_ON: S1Light > 100
- S1Light_OFF: S1Light < 101
- S2Light_ON: S2Light > 100
- S2Light_OFF: S2Light < 101
- S3Light_ON: S1Light > 200
- S3Light_OFF: S1Light < 201

"Lights Total" would be the sum of lights turned on.

CO2 variation: variation of the levels of CO2 in the lab classroom

- Decreasing Fast: z in [-1, 0.5]
- Decreasing: z in [-1, 0]
- Stable: z in [-0.3, 0, 0.3]
- Increasing: z in [0, 1]
- Increasing Fast: z in [0.5, 1]

The normalization of the variation was made using the equation below:

$$z = \frac{x}{\max(|minCO2|, maxCO2)}$$

After intensive testing we figured that a lot of this features weren't helpful regarding this problem, so we ended not using them.

While we were testing, we noticed that, almost every time, when a student used the classroom, he/she turned on the light of the desk so we used the light as an indicator of how many persons are inside the classroom. Assuming that the amount of lights turned on were the same as the number of persons inside the classroom, the features we chose to keep were "Lights Total" (with a slight difference shown below), that required all three light sensor features, and "Weather", that required "S3Light" (because it's the light sensor closest to the window) and "CO2" (in order to know when the room is empty). All the other features were discarded.

Lights Total: Amount of lights turned on using the light sensors ("S1Light", "S2Light", "S3Light").

- S1Light_ON: S1Light > 100
- S1Light_OFF: S1Light < 110
- S2Light_ON: S2Light > 100
- S2Light_OFF: S2Light < 110
- S3Light_ON: S1Light in [200, 300] & "cloudy"
- S3Light_OFF: S1Light < 201
- S3Light_ON: S1Light > 300 & "sunny"

With this change, we can assure that in a sunny day, the desk can be empty but receiving a lot of sunlight.

Rules

We started by estimating if S1Light and S2Light were, because this were independent of the "Weather" feature. We considered that the light was on if its value was over 100.

L1&L2	L2(0)	L2(1)
L1 (0)	0	1
L1 (1)	1	2

```
rule1 = ctrl.Rule(antecedent=((light_1["off"] & light_2["off"])), consequent=lights_12["0"], label="rule 1")
rule2 = ctrl.Rule(antecedent=((light_1["on"] & light_2["off"]) | (light_1["off"] & light_2["on"])), consequent=lights_12["1"], label="rule 2")
rule3 = ctrl.Rule(antecedent=((light_1["on"] & light_2["on"])), consequent=lights_12["2"], label="rule 3")
```

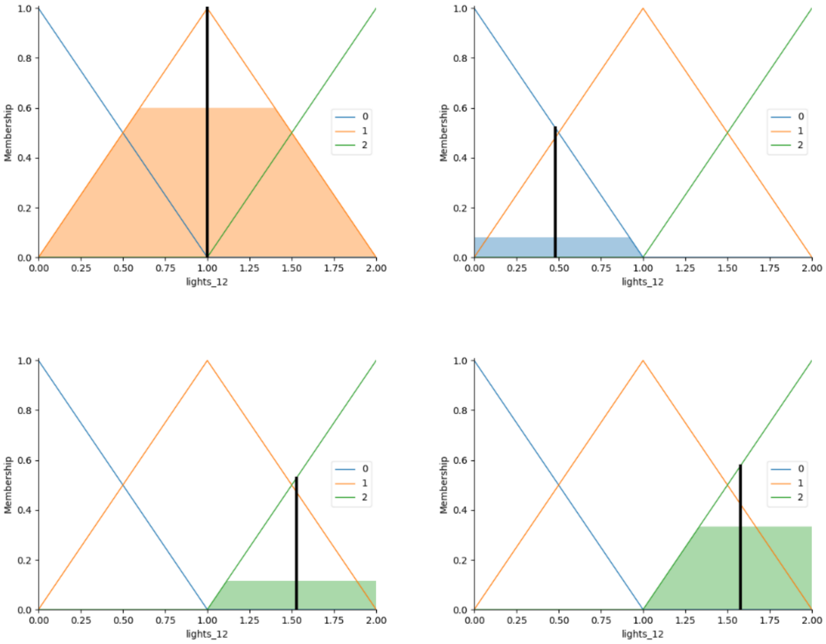

Figure 1: Representation of rules 1 to 3 (lights 1 and 2 on/off) with inputs (S1Light=100, S2Light=400), (S1Light=100, S2Light=230), (S1Light=270, S2Light=400), (S1Light=300, S2Light=230), respectively.

Then we created the "Weather" feature by comparing the value of "S3Light" with "CO2" value, in order to find if the classroom was full of light without having students inside. If the "CO2" value is above the limit and the light is still weak we can say that it's also a cloudy day. As for when there's both "S3Light" and "CO2" values above the limit, we considered that the worse case scenario was that it was a sunny day, this way we wouldn't consider that there was a student when the table was well illuminated without having a student on it.

Weather	C02 (< 400)	C02 (> 400)
L3(0)	Cloudy	Cloudy
L3(1)	Sunny	Sunny

```
rule4 = ctrl.Rule(antecedent=((light_3["low (w)"] & co2["lab empty"])), consequent=weather["cloudy"], label="rule 4")
rule5 = ctrl.Rule(antecedent=((light_3["high (w)"] & co2["lab empty"])), consequent=weather["sunny"], label="rule 5")
rule6 = ctrl.Rule(antecedent=((light_3["low (w)"] & co2["lab not empty"])), consequent=weather["cloudy"], label="rule 6")
rule7 = ctrl.Rule(antecedent=((light_3["high (w)"] & co2["lab not empty"])), consequent=weather["sunny"], label="rule 7")
```

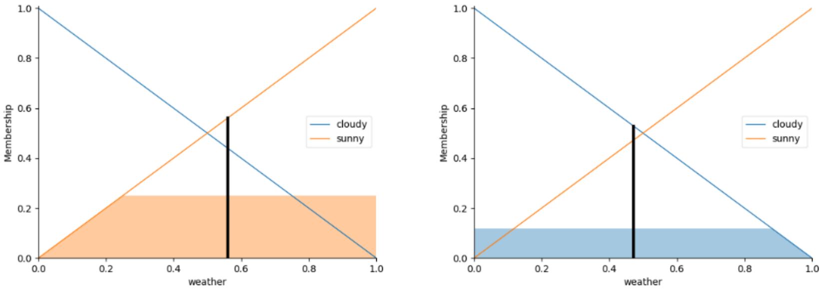

Figure 2: Representation of rules 4 to 7 (weather cloudy/sunny) with inputs (S3Light=245, CO2=300) and (S3Light=150, CO2=300), respectively.

Having the “Weather” feature created, we can now estimate if “S3Light” is detecting values of a lamp turned on, or if it is just sun light. For this we used three values for “S3Light” values: “low”, “medium” and “high”.

L3_final		Weather	
		Cloudy	Sunny
L3_inicial	Low	0	0
	Medium	1	0
	High	1	1

```
rule8 = ctrl.Rule(antecedent=light_3["low"], consequent=light3_f["0"], label="rule 8")
rule9 = ctrl.Rule(antecedent=(light_3["medium"] & weather_input["cloudy"]), consequent=light3_f["1"], label="rule 9")
rule10 = ctrl.Rule(antecedent=(light_3["medium"] & weather_input["sunny"]), consequent=light3_f["0"], label="rule 10")
rule11 = ctrl.Rule(antecedent=light_3["high"], consequent=light3_f["1"], label="rule 11")
```

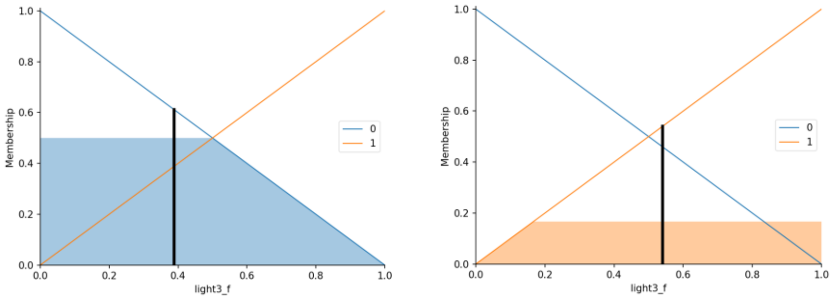

Figure 3: Representation of rules 8 to 11 (Light 3 on/off) with inputs (S3Light=200, Weather=cloudy) and (S3Light=350, Weather=cloudy), respectively.

After all lights are estimated, the only thing that is missing is merging this data with the data we got from the first set of rules (Lights 1 and 2).

Lights Total		L3	
		0	1
L1&L2	0	0	1
	1	1	2
	2	2	3

```
rule12 = ctrl.Rule(antecedent=lights_12_input["0"] & light_3_input["0"], consequent=total_lights["0"], label="rule 12")
rule13 = ctrl.Rule(antecedent=(lights_12_input["1"] & light_3_input["0"]) | (lights_12_input["0"] & light_3_input["1"]), consequent=total_lights["1"], label="rule 13")
rule14 = ctrl.Rule(antecedent=(lights_12_input["2"] & light_3_input["0"]) | (lights_12_input["1"] & light_3_input["1"]), consequent=total_lights["2"], label="rule 14")
rule15 = ctrl.Rule(antecedent=lights_12_input["2"] & light_3_input["1"], consequent=total_lights["3"], label="rule 15")
```

```
rule16 = ctrl.Rule(antecedent=total_lights_input["less than three"], consequent=room_overcrowded["0"], label="rule 16")
rule17 = ctrl.Rule(antecedent=total_lights_input["three"], consequent=room_overcrowded["1"], label="rule 17")
```

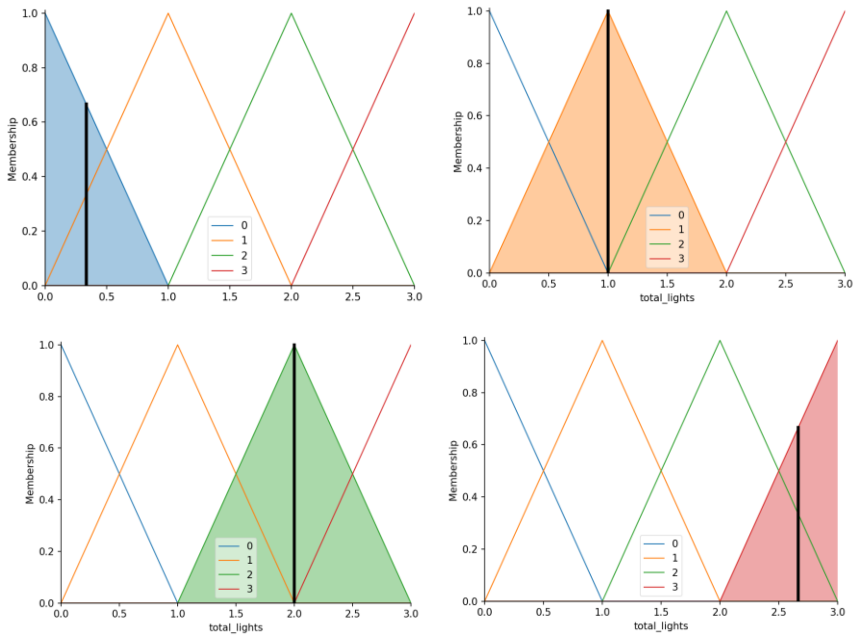

Figure 4: Representation of rules 12 to 17 (Amount of lights on/off) with inputs (Lights1_2=0, Light3=0), (Lights1_2=0, Light3=0), (Lights1_2=0, Light3=1), (Lights1_2=2, Light3=0), (Lights1_2=2, Light3=1), respectively.

Having this combination, we can now "know" when there's three lights on inside the classroom, considering this as an indicator of having more than two students inside the classroom.

In theory, this would work. However, after reaching the end we couldn't get good results. Using the dataset, our Fuzzy System couldn't detect a single record with the classroom with more persons than the allowed. We tried to fix our code but we weren't successful.

As we could get a reasonable output using the dataset we didn't made the confusion matrix or the comparison with the NN-model.