Group project assignment

Knowledge Representation

The group project consists of several programming tasks combining Prolog (second module) and cognitive mapping modeling (third module). Students are required to create (1) a Prolog knowledge base and code (2) some Prolog scripts to (3) answer queries using the symbolic simulation results. You will receive numerical simulations (inputs and outputs) produced with a Fuzzy Cognitive Map model concerning business intelligence. In addition, you will receive the results of these simulations expressed as symbolic terms. Please notice that you do not have to create the map or run the simulations yourself, yet we will provide you with a comprehensible explanation about the model and the simulation process. We expect you upload a report explaining your solutions. This programming assignment counts for 20% of the final grade.

Model description

In this section, we will explain the Fuzzy Cognitive Map (FCM) model used to perform the simulations. Figure 1 shows an FCM-based concerning business intelligence. This model involves seven concepts: *E-Business Profits* (C1), *E-Business Sales* (C2), *Prices Cutoffs* (C3), *Customers' Satisfaction* (C4), *Staff Recruitment* (C5), *Impact from International E-Business Competition* (C6), and *Better E-Commerce Services* (C7). In addition, we can notice that there are several negative and positive causal relationships attached to the network, yet the graph is not fully connected (not every concept is connected with the others).

In the previous model, there is no distinction between the concepts when it comes to inputs and outputs. This means that all concepts will be used as inputs and outputs when performing what-if simulations.

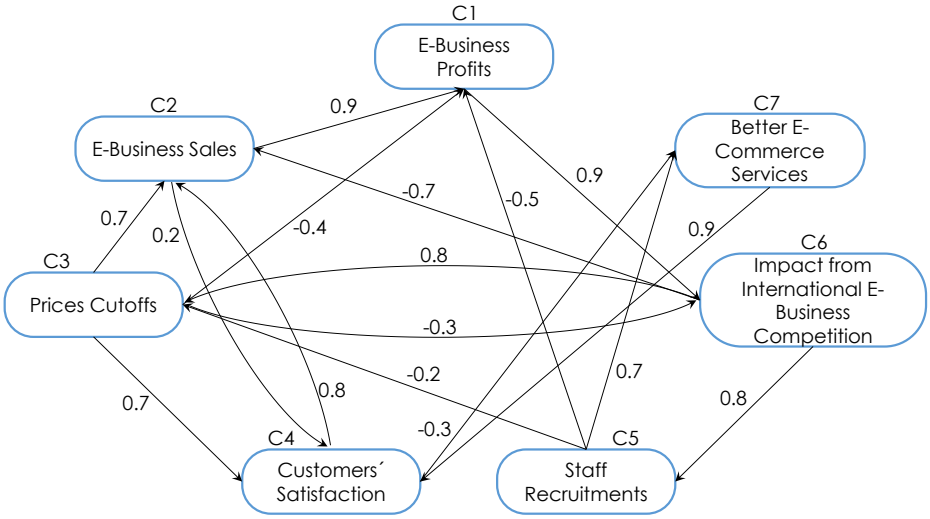

Figure 1: FCM-based model concerning business intelligence.

The system has been designed by an expert panel, so we do not argue about the accuracy of the causal weights connecting the concepts.

The expert panel informed us that the distribution of variables resembles the behavior of the sigmoid function (see next Figure 2). This means that we should use the sigmoid transfer function when performing what-if simulations in the next section to generate the numerical dataset.

Unfortunately, we have bad news. After running some preliminary simulations, we noticed that the results did not change for different initial activation vectors. This means that the FCM converged to a *unique* fixed-point attractor, so the simulation results are pretty much worthless.

If the FCM model converges to a unique fixed-point attractor, the model will produce the same output regardless of the inputs. Unfortunately, there is not much we can do about it without changing the model.

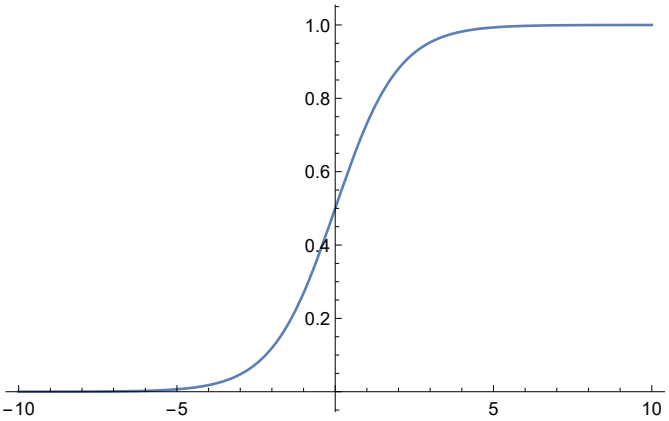

Figure 2: Sigmoid transfer function without any transformation.

We have approached the expert panel to discuss whether some changes could be altered (hoping we can avoid the unique fixed-point attractor). However, the experts are reluctant to do that. They fairly claim that altering the weight matrix would harm the underlying semantics of the model.

After a brief panic moment, we decided to modify the inference mechanism combining all information we have about the concepts' states. The new modification combines the initial and previous states of each concept with the activation flow that comes from connected concepts. Equation (1.1) formalizes the modified reasoning mechanism developed by the lecturers,

$$a_i^{(t)} = \frac{1}{2} f\left(a_i^{(t-1)} + a_i^{(0)}\right) + \frac{1}{2} f\left(\sum_{j=1}^{M} w_{ji} a_j^{(t-1)}\right) \tag{1.1}$$

where $a_i^{(t)}$ is the activation value of the i-th concept in the t-th iteration, M represents the number of neural concepts in the network, while $f(\cdot)$ is the sigmoid transfer function shown in Figure 2. In addition, we standardized the concepts' activation values to prevent a poor covering of the activation space (please refer to the Covering section from the N-book).

Note. Students are not expected to know the details explained in this section. Such details were included in the assignment to awake your curiosity and illustrate why mastering the theory is pivotal in this course.

Although the above equation looks scary, its intuition is quite straightforward. The process can be summarized as follows. Firstly, each neuron is activated using a certain *initial activation value*, which belongs to the allowed activation interval. The expert determines the initial activation values and which neurons to activate. Secondly, we perform reasoning such that the output of neurons in the previous iteration (after being processed by the transfer function) is used as the input of the following iteration. If we arrange all activation values in a vector, then the inference process will consist of chained vector-matrix multiplications. Overall, the stimuli concern the IF part of the simulation, while the outputs (after a fixed number iterations) concern the WHAT part.

We are aware that some students prefer to align the math with concrete code implementations. Below you can find the Python code implementing the new reasoning procedure for FCM-based models.

```
import numpy as np

def sigmoid(A, l=2.5, h=0.5):
    return 1.0 / (1.0 + np.exp(-l*(A-h)))

def reasoning(W, A, T=50, alpha=0.5):

    states = np.zeros([len(A), T, len(W)])
    states[:,0,:] = A = standardize(A)

    for t in range(1,T):
        A = alpha * sigmoid(A + states[:,0,:]) + (1-alpha) * sigmoid
    (np.matmul(A, W))
        states[:,t,:] = A = standardize(A)

    return A

def standardize(A):
    for i in range(A.shape[1]):
        A[:,i] = (A[:,i] - A[:,i].min())/(A[:,i].max() - A[:,i].min
    ())
    return A
```

Simulation results

In order to perform what-if simulations, we generated 20 random vectors with values in the $[0, 1]$ interval. Each randomly generated vector has length equal to seven since all concepts are regarded input one.

Table 1 shows the inputs we will use to feed the model and perform the simulations. Each row denotes an input vector, which can be understood as an initial activation vector used to start the reasoning process.

Table 1: Initial activation vectors (numerical inputs).

	C1	C2	C3	C4	C	C6	C7
I1	0.55	0.71	0.29	0.51	0.89	0.9	0.13
I2	0.21	0.05	0.44	0.03	0.46	0.65	0.28
I3	0.68	0.59	0.02	0.56	0.26	0.42	0.28
I4	0.69	0.44	0.16	0.54	0.78	0.31	0.22
I5	0.39	0.94	0.98	0.67	0.9	0.85	0.38
I6	0.09	0.65	0.56	0.36	0.23	0.41	0.47
I7	0.27	0.29	0.46	0.86	0.59	0.28	0.28
I8	0.45	0.21	0.2	0.51	0.09	0.48	0.36
I9	0.71	0.75	0.69	0.69	0.37	0.67	0.34
I10	0.57	0.33	0.45	0.06	0.24	0.97	0.23
I11	0.69	0.65	0.72	0.48	0.6	0.07	0.07
I12	0.2	0.15	0.1	0.13	0.55	0.19	0.95
I13	0.68	0.54	0.71	0.26	0.93	0.84	0.73
I14	0.48	0.84	0.74	0.66	0.91	0.63	0.37
I15	0.55	0.2	0.19	0.73	0.78	0.97	0.85
I16	0.54	0.09	0.49	0.93	0.79	0.49	0.46
I17	0.22	0.18	0.07	0.89	0.64	0.14	0.41
I18	0.05	0.21	0.73	0.65	0.48	0.27	0.65
I19	0.96	0.44	0.07	0.06	0.08	0.96	0.54
I20	0.84	0.17	0.26	0.69	0.9	0.34	0.06

Table 2 shows the simulation results produced by the FCM-based model after performing $T = 50$ iterations. Please be aware that those activation vectors correspond to the concepts' activation values in the last iteration. The intermediate activation vectors were discharged for the sake of simplicity.

Table 2: Final activation vectors (numerical outputs).

	C1	C2	C3	C4	C5	C6	C7
I1	0.83	0.88	0.86	0.85	1	0.93	0.45
I2	0	0.09	0.76	0.15	0.56	0.49	0.69
I3	0.94	0.74	0.58	0.81	0.66	0.92	0.53
I4	0.89	0.77	0.63	0.86	0.92	0.8	0.64
I5	0.77	1	1	0.99	0.98	0.87	0.84
I6	0.77	0.95	0.96	0.82	0.12	0.71	0.51
I7	0.72	0.87	0.78	0.93	0.72	0.54	0.53
I8	0.91	0.61	0.9	0.74	0	0.87	0.19
I9	0.92	0.97	1	0.98	0.82	0.92	0.71
I10	0.76	0.46	0.96	0.27	0.57	0.89	0.55
I11	1	0.94	0.8	0.61	0.6	0.36	0
I12	0.25	0	0	0	0.25	0	0.89
I13	0.85	0.88	0.98	0.84	1	0.92	1
I14	0.83	0.99	0.97	0.98	0.97	0.86	0.83
I15	0.76	0.65	0.77	0.99	0.98	0.92	0.99
I16	0.76	0.65	0.88	1	0.91	0.77	0.85
I17	0.57	0.43	0.08	0.77	0.58	0.28	0.69
I18	0.53	0.91	0.79	0.94	0.33	0.26	0.71
I19	0.95	0.49	0.84	0.28	0.12	1	0.68
I20	0.83	0.53	0.73	0.75	0.92	0.76	0.21

The numerical simulations are very useful when it comes to sub-symbolic reasoning since they provide accuracy. However, they might be too detailed. On the one hand, experts might not be interested in the exact numerical values. Instead, it would be more appealing to have symbolic terms describing each concept state. On the other hand, numerical values might not be the ideal granularity degree when designing Prolog-based solutions.

To transform numerical inputs and outputs into symbolic ones, we used *fuzzy logic*. Firstly, we define six linguistic terms: *near zero* (z), *very low* (vl), *low* (l), *medium* (m), *high* (h) and *very high* (vh). Secondly, we define a Gaussian membership function for each fuzzy set. Figure 3 shows these functions (we will exclude the equations for the sake of simplicity). Finally, we evaluate each numeric value with each function such that we can determine which fuzzy set is more suitable for that value (that is to say, the fuzzy set with the highest membership degree). This process will be more clear after the practicals.

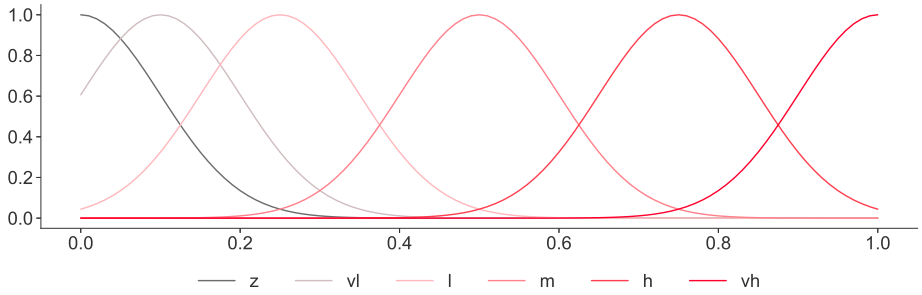

Figure 3: Gaussian membership functions.

Tables 3 and 4 show the symbolic input and output values after performing the fuzzification process on the numerical inputs and outputs, respectively. Please notice that there is a number next to each symbolic term. This number represents the membership value related to that symbolic term and can be understood as the confidence of that symbolic term.

Table 3: Symbolic inputs after the fuzzification process.

	C1		C2		C3		C4		C5		C6		C7	
I1	m	0.88	h	0.92	l	0.96	m	1.00	vh	0.55	vh	0.61	vl	0.96
I2	l	0.92	z	0.88	m	0.84	z	0.96	m	0.92	h	0.61	l	0.96
I3	h	0.78	m	0.67	z	0.98	m	0.84	l	1.00	m	0.73	l	0.96
I4	h	0.84	m	0.84	vl	0.84	m	0.92	h	0.96	l	0.84	l	0.96
I5	m	0.55	vh	0.84	vh	0.98	h	0.73	vh	0.61	h	0.61	m	0.49
I6	vl	1.00	h	0.61	m	0.84	l	0.55	l	0.98	m	0.67	m	0.96
I7	l	0.98	l	0.96	m	0.92	h	0.55	m	0.67	l	0.96	l	0.96
I8	m	0.88	l	0.92	l	0.88	m	1.00	vl	1.00	m	0.98	l	0.55
I9	h	0.92	h	1.00	h	0.84	h	0.84	l	0.49	h	0.73	l	0.67
I10	m	0.84	l	0.73	m	0.88	vl	0.92	l	1.00	vh	0.96	l	0.98
I11	h	0.84	h	0.61	h	0.96	m	0.98	m	0.61	vl	0.96	vl	0.96
I12	l	0.88	vl	0.88	vl	1.00	vl	0.96	m	0.88	l	0.84	vh	0.88
I13	h	0.78	m	0.92	h	0.92	l	1.00	vh	0.78	h	0.67	h	0.98
I14	m	0.98	h	0.67	h	1.00	h	0.67	vh	0.67	h	0.49	l	0.49
I15	m	0.88	l	0.88	l	0.84	h	0.98	h	0.96	vh	0.96	h	0.61
I16	m	0.92	vl	1.00	m	1.00	vh	0.78	h	0.92	m	1.00	m	0.92
I17	l	0.96	l	0.78	vl	0.96	vh	0.55	h	0.55	vl	0.92	m	0.67
I18	z	0.88	l	0.92	h	0.98	h	0.61	m	0.98	l	0.98	h	0.61
I19	vh	0.92	m	0.84	vl	0.96	vl	0.92	vl	0.98	vh	0.92	m	0.92
I20	h	0.67	vl	0.78	l	1.00	h	0.84	vh	0.61	l	0.67	vl	0.92

Table 4: Symbolic outputs after the fuzzification process.

	C1		C2		C3		C4		C5		C6		C7	
I1	h	0.73	vh	0.49	h	0.55	h	0.61	vh	1.00	vh	0.78	m	0.88
I2	z	1.00	vl	1.00	h	1.00	vl	0.88	m	0.84	m	1.00	h	0.84
I3	vh	0.84	h	1.00	m	0.78	h	0.84	h	0.67	vh	0.73	m	0.96
I4	vh	0.55	h	0.98	h	0.49	h	0.55	vh	0.73	h	0.88	h	0.55
I5	h	0.98	vh	1.00	vh	1.00	vh	1.00	vh	0.98	h	0.49	h	0.67
I6	h	0.98	vh	0.88	vh	0.92	h	0.78	vl	0.98	h	0.92	m	1.00
I7	h	0.96	h	0.49	h	0.96	vh	0.78	h	0.96	m	0.92	m	0.96
I8	vh	0.67	m	0.55	vh	0.61	h	1.00	z	1.00	h	0.49	l	0.84
I9	vh	0.73	vh	0.96	vh	1.00	vh	0.98	h	0.78	vh	0.73	h	0.92
I10	h	1.00	m	0.92	vh	0.92	l	0.98	m	0.84	vh	0.55	m	0.88
I11	vh	1.00	vh	0.84	h	0.88	m	0.55	m	0.61	l	0.55	z	1.00
I12	l	1.00	z	1.00	z	1.00	z	1.00	l	1.00	z	1.00	vh	0.55
I13	h	0.61	vh	0.49	vh	0.98	h	0.67	vh	1.00	vh	0.73	vh	1.00
I14	h	0.73	vh	1.00	vh	0.96	vh	0.98	vh	0.96	h	0.55	h	0.73
I15	h	1.00	h	0.61	h	0.98	vh	1.00	vh	0.98	vh	0.73	vh	1.00
I16	h	1.00	h	0.61	vh	0.49	vh	1.00	vh	0.67	h	0.98	h	0.61
I17	m	0.84	m	0.78	vl	0.98	h	0.98	m	0.78	l	0.96	h	0.84
I18	m	0.96	vh	0.67	h	0.92	vh	0.84	l	0.73	l	1.00	h	0.92
I19	vh	0.88	m	1.00	h	0.67	l	0.96	vl	0.98	vh	1.00	h	0.78
I20	h	0.73	m	0.96	h	0.98	h	1.00	vh	0.73	h	1.00	l	0.92

The symbolic inputs and outputs resulting from the defuzzification step (together with their confidence values) are all we need to build a Prolog knowledge base and perform symbolic reasoning. This is exactly what you have to do in this assignment. More details will follow in the next section.

Note. Students are not expected to master the details behind the simulation and knowledge transformation processes. That is not mandatory to solve the assignment successfully. Moreover, these steps will be clearly explained during the sessions belonging to the third module (sub-symbolic knowledge representation). However, we did not want to give you the symbolic inputs and outputs without detailing how those symbolic terms were obtained.

Assignment details

In this section, we will explain the problems you have to solve using the symbolic data generated in the previous section. Those problems must be solved using the Prolog contents discussed during the course.

Firstly, students are requested to create a Prolog knowledge base using the symbolic inputs and outputs (together with their confidence values) provided in Tables 3 and 4, respectively. Secondly, students are requested to code Prolog scripts and queries answering the following tasks and include their Prolog scripts and queries under their answers to the following questions:

1. Determine all output values of *Staff Recruitment* and *Better E-Commerce Services* when *E-Business Profits* and *E-Business Sales* are initialized with high (h) and very high (vh), respectively. The remaining variables are not relevant to this simulation. Please return the confidence values of each symbolic solution. Which solution you prefer and why? (max 3 sentences, excluding the Prolog code and outputs)

2. Modify the solution to the first task to ensure that the confidence values of solutions for *Staff Recruitment* and *Better E-Commerce Services* are greater than 0.8. Explain two advantages of this modification. (max 3 sentences per advantage, excluding the Prolog code and outputs)

3. Which initial values should the variables *Customers' Satisfaction* and *Impact from International E-Business Competition* have taken such that the variables *Staff Recruitment* and *Better E-Commerce Services* can produce high activation values? Why is this simulation useful? (max 5 sentences, excluding the Prolog code and outputs)

In order to solve the previous tasks, you might need to manipulate the knowledge base dynamically. This can be done using the following predicates: `dynamic` [1], `assertz` [2] and `retractall` [3]. The first predicate informs the Prolog interpreter that the definition of the predicate(s) may change during execution (using `assertz` and/or `retractall`). The second predicate asserts a clause (fact or rule) into the database. Finally, the third predicate removes all facts or clauses for which the head unifies the specified head. The way you will use these predicates depends entirely on the solutions to the tasks.

[1] https://www.swi-prolog.org/pldoc/man?predicate=dynamic/1
[2] https://www.swi-prolog.org/pldoc/man?predicate=assertz/1
[3] https://www.swi-prolog.org/pldoc/man?predicate=retractall/1

Report. Students are requested to submit a report with the solution of the assignment (including the code solutions and the knowledge base). The report must be uploaded to Canvas as a single PDF file. The report must meet some minimal requirements (e.g., the solution is properly explained, the questions must be concisely answered). The report must not exceed three pages.

Grade. The project's grade will be determined in a binary way: pass or no-pass. In order to get a pass, the report must meet the minimal requirements AND the Prolog programs must run. This means that we will evaluate the report as a whole, taking into account clarity, explanations, assumptions and the implementation. Students can expect a brief feedback report on their projects with the weakest and strongest points. Students do not need to pass the group assignment in order to pass the course successfully. Moreover, there will be a resit opportunity for those groups that fail the first submission attempt.

Deadlines. There are several deadlines related to each submission (i.e., regular and resit). The timeline with the deadlines for the regular project submission is depicted below. The resit deadlines will be released later.

02.03.2021	The project assignment is released on Canvas
13.04.2021	Technical Q&A session with the lecturers
26.05.2021	The registration form is uploaded to Canvas
01.06.2021	The project report is uploaded to Canvas
15.06.2021	The pass/no-pass decisions are released

Have a lot of fun!!