[Re] The Wisconsin Card Sorting Test: Theoretical analysis and modeling in a neuronal network

Pauline Bock[1,2,3] and Frédéric Alexandre[2,1,3]

[1]LaBRI, Université de Bordeaux, Institut Polytechnique de Bordeaux, Centre National de la Recherche Scientifique, UMR 5800, Talence, France – [2]INRIA Bordeaux Sud-Ouest, Bordeaux, France – [3]IMN, Université de Bordeaux, CNRS, UMR5293, Bordeaux, France

1 Introduction

The Wisconsin Card Sorting Test is a psychological test used to evaluate cognitive flexibility capacities in patients with decision-making deficits, to evaluate among other things, some mechanisms dysfunctions of frontal lobe. In the test it is proposed to match several cards, composed of several colored forms, by the 3 different criteria: color, figures or quantity, according to 4 reference cards (Heaton, R.K, (1981))[1]. The subject is told to match the cards, but the rule must be found by trials and errors and might change without notice. The ability of the subject to adapt to the changes is measured in the test. In 1991, S. Dehaene and J.P. Changeux have created a neuronal network model to resolve Wisconsin Card Sorting Test (Dehaene & Changeux, 1991)[2], enhancing 3 cognitive components necessary to pass the test:

- the ability to change a followed strategy/rule when an error occurs

- the ability to memorise wrong strategies to avoid repeating them

- the ability of auto-evaluation, i.e comparing different situations and their outcomes referring to a memory of the past.

This model refers to some biological mechanisms and their supposed implementation in the brain circuitry, and is interesting to model some flexible processes used in decision-making and executive control. Although this model has been published in 1991, no implementations have been shared. We have implemented this model in Python with some new mechanisms, trying to reach the author's theoretical results.

2 Method

2.1 Model of Dehaene and Changeux

The model, depicted below in figure 1, is composed of several layers of neuronal units that are called clusters to emphasize the fact that each unit does not implement the processing of a single neuron but rather of a population of neurons. Each unit or cluster is activated by a sigmoid function whose formula is available in annexes. Clusters of neurons have only 2 activity states: either they are active or they are inactive. Their activity

varies between 0 and 1, the switch from a state to another being defined by a threshold fixed at 0.5 by the authors. The layers are linked by some connections composed of a multiplication of a short-term and a long-term component. A short-term component represents heterosynaptic influences on a connection between 2 clusters, coming from another third modulator neuron: it is what the authors calls a "synaptic triad". This component varies between 0 and 1. A long-term component varies between 0 and a maximum value, representing direct influence from a layer to another.

For one test trial, i.e a response card to match to one of the reference cards, the order of activation of the layers is the following: first, the response cards features are coded in the input layer, they are separated in 3 assemblies, one for each dimension. The input is then connected to a memory layer, to keep the card features in mind during auto-evaluation. There is a competition mechanism for each assembly to ensure that only one feature is activated per dimension.

Then memory is partially connected to an intention layer. In these connections are coded the features of the 4 reference cards. These connections are modulated by the rule-coding layer clusters. Only one rule can be activated at a time by competition and will increase the connections of the dimension assembly associated, while the others will be decreased, influencing the activation of an intention. A mechanism of competition will then ensure that only one intention is activated at a time.

The intention layer is connected to an output layer and to an error cluster. A go unit modulates the intention-to-output connections to wait before making an action. In the paper, this unit is activated externally without more details. The connections from intention to the error cluster is where the auto-evaluation component is encoded. Indeed, the error cluster can be either activated by the reward input (when the choice lead to an error) or by the intention activities. This error cluster, when activated, will change the rule-coding layer in what the authors call a "generator of diversity", i.e the auto-excitatory connection of the rule cluster activated will be depressed until the rule cluster totally deactivates because of the inhibition of other rule clusters. The competition will activate an other rule. The memory of rejected rules is coded in a recovery rate of each rule that will make the rule become competitive again more or less quickly.

2.2 Implementation and modifications

The model was implemented with Python and Numpy library. Several matrices were implemented for the short-term and long-term components of the connections, and for the activities of the layers. They were initialised with the values given in the paper, and updated with the formulas given there too.

Some elements have then been added to get closer to the theoretical results reported in the paper, as a simple implementation of the components described in the paper was not sufficient to have the same results. Indeed, there were some problems of memory to be maintained while auto-evaluation, or with the activation of the go unit. In fact, the behaviour of the model could be good enough but the algorithms used to activates the go unit were not inspired by some biological mechanisms. As said before, when the go unit is activated, the intention is transmitted to the output units. In the original paper, it is written that the go unit is activated by an external signal and the following comment is given: "The go cluster might also be activated by endogenous decision processes, but we do not use this possibility in the following simulations". Consequently, in the reported simulations in the original paper, the go unit was triggered by the programmer. What we did in this paper is to add some neuronal units to implement the mentioned endogenous process and have, without an external signal, the same experimental results.

3 units have been added:

- a neuronal cluster of confidence, whose role is to compute the error activity "inverse", its activity is high when error activity is low and low when error activity is
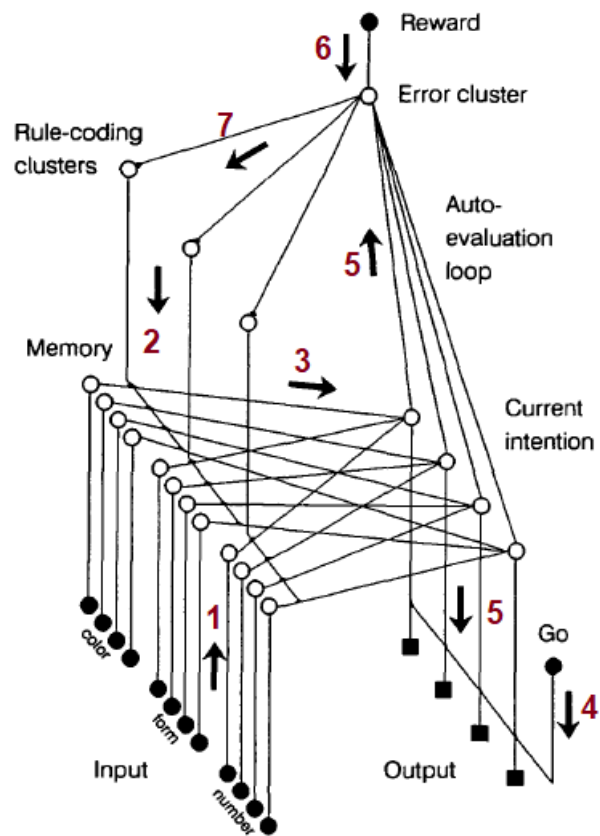
Figure 1. Schematic architecture of the neuronal network model of Dehaene and Changeux coming from their article. The approximative order of execution of the network is indicated by numbers.

high.

- a neuronal cluster of "reflexion" controlling response card inputs within memory layer, in order to have enough time for auto-evaluation according to previous card.

- a neuronal cluster of "inhibition", controlling go and reflexion units activation, delaying motor output and action choice while memory is changing.

The new model architecture is shown in the figure 2:

2.3 Model functioning

Firstly, all neuronal activities are set to 0 before network activation, except for one rule-coding cluster, whose activity is set near 1, as the first decision is supposed to be random at the test begining, having no a priori about which rule to sort by.

Then, the connection components and activity thresholds are initialised with different values. These values are the ones for which the model has the best stability, dynamics and neuron activity behaviors. The initialised values and formulas of activity and connection updates computations are available in Annexes.

In order to evaluate the auto-evaluation ability, the authors have created a modified version of the Wisconsin Card Sorting Test, using only 36 ambiguous cards, i.e response cards that match a reference card according to 2 dimensions. The rule changes whenever the agent made three consecutive correct assignment according to the current right rule, meaning a criterion is reached. The test is finished when 6 criteria are reached
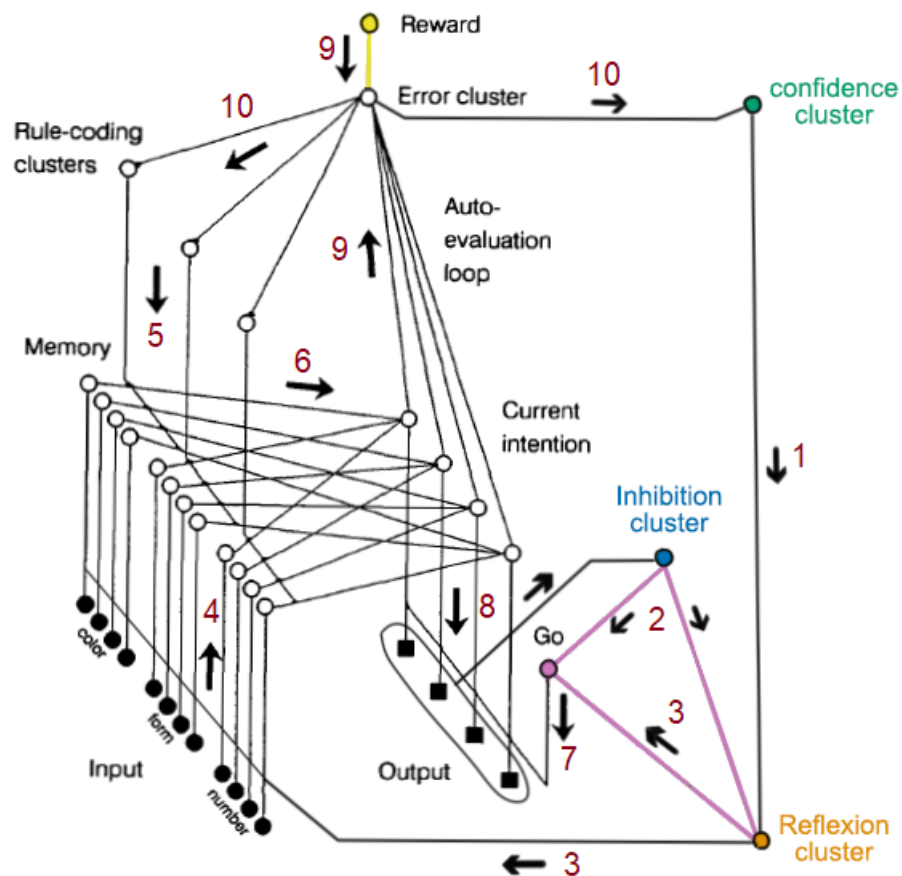
Figure 2. Modified schematic architecture adapted from S.Dehaene and JP.Changeux model (1991). The approximative order of execution of the network is indicated by numbers. From basic model are added 3 units. An inhibition unit (blue) activates when one output is active. When active, the inhibition unit inhibits the go and reflexion units, and the output inhibits itself, delaying motor output. The reflexion unit (orange) modulates input-to-memory synapses: when active, the new response card is transmitted to memory, and when inactive, the input of the new card into memory is blocked. The confidence unit (green) activates reflexion unit when error activity is low, indicating that an a priori confident choice has been done and that a new decision can be taken according to the new card. An internal reward system have been added (in yellow) and an internal mecanism of activation of the go unit (in purple).

(color-shape-quantity twice) or when all the 36 cards have been used. So, while the 6 criteria are not reached , the network activity is computed.
The order of unit activation within a trial is as follows: inhibition and confidence units are computed, followed by reflexion: it allows to know if we are in a auto-evaluation phase on the previous card or in an active phase to choose a reference card to sort the new response card accordingly.

Memory is then updated or not according to reflexion activity, and transmitted to intention according to rule-coding clusters modulation on the connections. Then, go unit activates or not according to reflexion activity. When go unit is activated, the intention is transmitted to the output. Intention is as well transmitted to the error cluster for the auto-evaluation part. The error is moreover influenced by a reward input. There are 2 types of reward possible: when at least one output cluster have an activity higher than 0.5, an action is made, and an external reward is received, equal to 1 if the choice was an error, to activates error cluster, or equal to 0 if the choice was right, not to influence
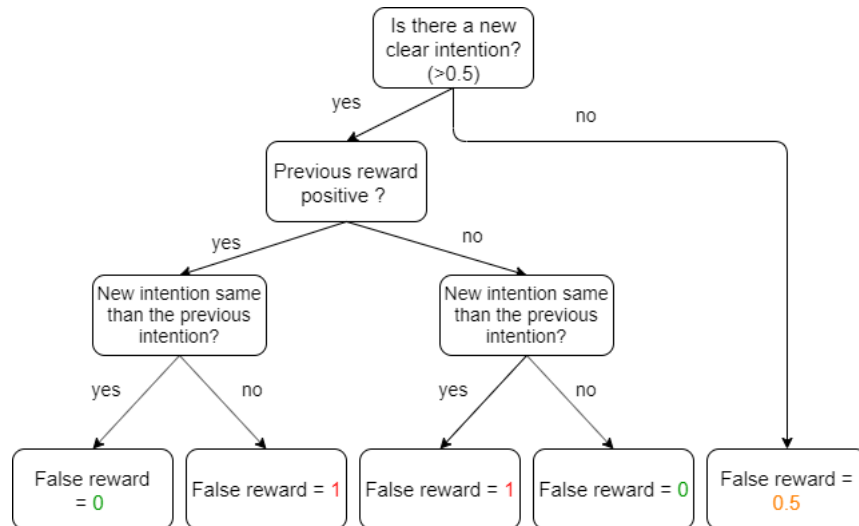
Figure 3. Tree diagram representing the value of the false reward attributed according to the previous reward and the intentions.

error activity. The other reward is one that has been added to the model. While reflex ion isn't active, a "fake" reward is computed in order to realise a better auto-evaluation on the past events.

Its value is depending on the previous external reward received, and on the previous card chosen that lead to this reward. This choice is compared to the new intention according to a new rule followed. 5 different events can exist according to the previous true reward from the last trial and the comparison between intentions as showed in the following figure 3.

Then the synaptic efficacy of the connections are updated: input-to-memory connections are modulated by reflexion activity, memory-to-intention short-term component by rule-coding cluster activities and long-term component by error, intention and memory activities according to an Hebbian learning rule. Intention-to-output short-term component are modulated according to go activity.

Rule-coding cluster auto-excitatory short-term components are updated according to error and self rule activities. Finally, output and inhibition auto-excitatory short-term components are updated. Output short-term components are modulated by output activity itself, allowing output activity not to stay active for too long, delaying motor output. Inhibition is updated by confidence activity, its auto-excitatory long-term component decreases when confidence is high, to let reflexion and go unit become activated, as a confident intention is done, and on the contrary the long-term component increases when error is high, to stay in a auto-evaluation phase.

2.4 Neuronal Network test

In order to evaluate the performance of the model according to the one described in the original paper, statistical computations have been done. It has been chosen to only implement and test the model with auto-evaluation and memory functioning, using modified WCST version, to evaluate these components as it has been done in the paper. The different computations that have been done are the following ones: the speed of learning, the one-trial learning rate and the perseveration rate. Speed of learning statistics are computed by increasing an error accumulator each time an external negative reward is received. It is saved in a list and re-initialised each time a criterion is reached. Then the mean of this list is computed in order to have the mean negative trial number necessary to reach a criterion.

The single-trial learning is the percentage of immediate succes obtained after only one negative trial. It is computed by counting the number of success following a negative trial (thanks to reward history) divided by the number of success plus the number of failure following one negative trial multiplied by 100.

The perseveration rate is computed as follows: if the previous trial was negative, and the actual trial also, if the active rule is the same as in the previous trial, then it is a perseveration error. These perseveration errors are counted and divided by the total number of errors, multiplied by 100.

A script has been implemented to compute these statistics on 500 tests like in the paper. Furthermore, the activities of some units have been saved and displayed at the end of a test, to evaluate the model behaviour according to different situations. Some of these graphs can be found in Annexes.

3 Compared results

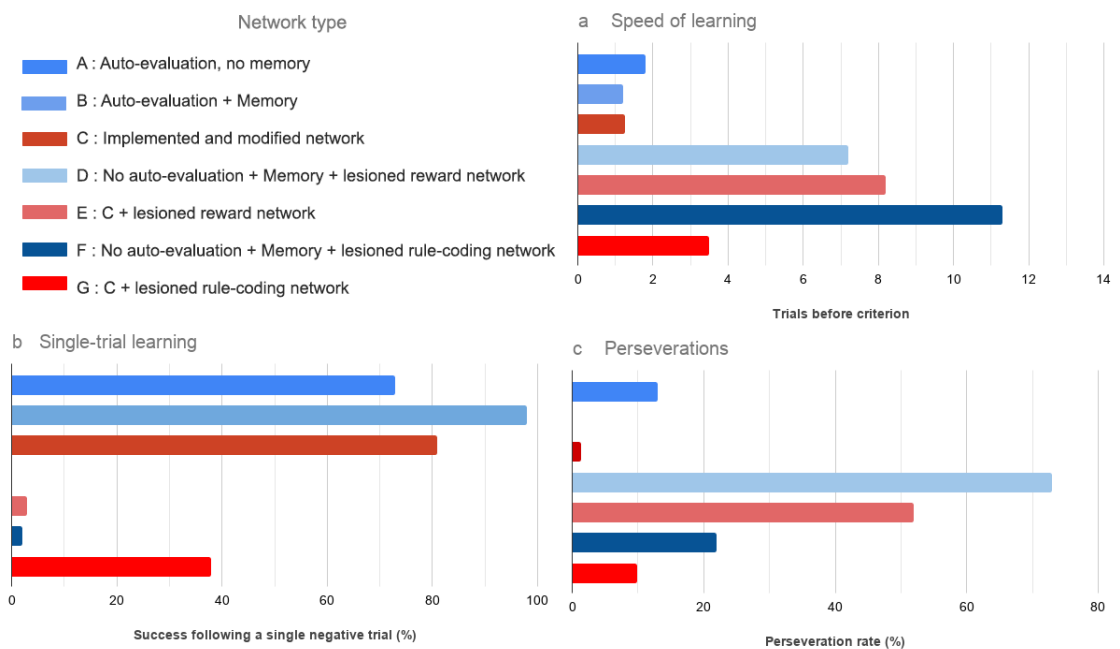The 500 test statistical computations have led to the results visible in figure 4.

Figure 4. Graphs of results adapted from the graph included in the original article of Dehaene and Changeux. *A comparison between the original model and our re-implementation: Each pattern in the bluescale bar graph represents a different model used in the original paper (see legend). The redscale bar graphs represent the value obtained with our re-implementation. The overlap in the bluescale and redscale bar graphs shows a direct comparison between the value obtained with our re-implementation and those obtained by the original model. Comparing with the model using auto-evaluation and memory components, the results of the implemented model are very similar to the results of the original model with auto-evaluation and/or memory. The exact values are the following for the optimal model: Speed of learning: 1.267, Single-trial learning: 80.393%, Perseverations rate: 1.272%. The modified model has been lesioned to compare results with the model without auto-evaluation and with lesioned reward and rule-coding networks.*

Some differences of results can be found between the optimal model of the original paper and the model implemented, but it can be explained by seeing activities of the clusters. For example, it can sometimes take 2 trials instead of one to immediately succeed, as sometimes when there is a negative trial, the auto-evaluation on the previous

card can lead to choose a different intention than the wrong one, following the 2 other rules, if the features of the response cards were pointing to the same card. As it is said in the paper, then the rule is believed to be the right one, because it has led to another intention than the wrong one, but in fact, it is the other rule that must have been chosen. An example of this event can be seen in one of the activity graphs reported in figure 5:
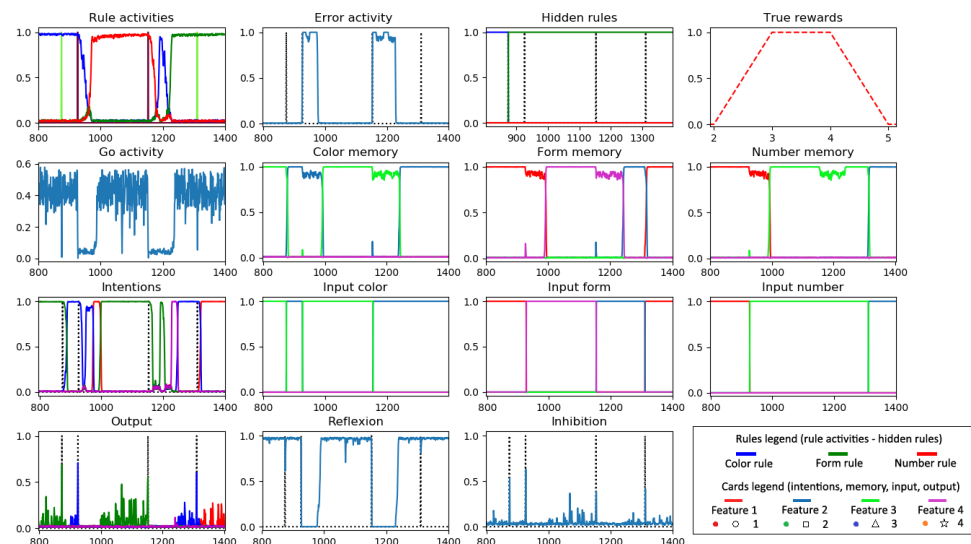

Figure 5. Recording of clusters activities made with matplotlib library and pyplot module. *The card legend colors represent each reference cards (card 1 red, card 2 blue, card 3 green and card 4 violet). Here, the graphs have been zoomed on some loops to show that 2 negative trials can be necessary before succeeding. In rule activities, one can see that color rule was activated, and then, the hidden rule is changing for shape/form rule. So, the model is making an error after the rule was changed, and the color rule is deactivated. The quantity/number rule is then tested during auto-evaluation. The color rule led to choose the second card in the previous trial (the blue curve in intention) according to the color memory (blue curve too) and the quantity rule leads to choose another card, that is the first one. So as this rule would lead to another card than the wrong previous one, the quantity rule keeps activated and a choice is made according to the new card in input and actualised memory. But in fact, it was the form rule that was the right one, but as these 2 rules would have led to choose the same different card than with following the color rule, an error is made.*

Other differences can be seen between the lesioned modified model and the lesioned model in the original article, the modified model has better results even when lesioned but the results are still worse than the ones for the no-lesioned model. When the modified model is lesioned, the test is no longer successful and only a mean of 1 or 3 criterions are reached instead of the 6 required to succeed.

4 Conclusion

An implementation of the model of Wisconsin Card Sorting Test of Dehaene and Changeux has been done, adding some modifications to try to reach the paper results for the model with auto-evaluation and memory abilities, as there was some lack of information or misunderstandings in the paper to reach these results without modifications. There can be some errors coming from model stability, but the results obtained are close to the paper ones concerning the optimal model. The modifications that have been done, compared to the model without modifications, have improved the basic model in terms of biological mechanisms, as the go unit is now activated by units that give a focus on perception and are influenced by a computation of confidence on the auto-evaluation.

5 Annexes

5.1 S.Dehaene and JP.Changeux model computations

For the new added units, the initialising values have been set respecting the order of magnitude of the paper values. Here are the fixed values at the beginning:

- Long-term: input-to-memory: 3, memory-to-intention: 3, intention-to-output: 3, intention-to-error: 5, reward-to-error: 6, inhibition-to-go: -6, reflexion-to-go: 3, output-to-inhibition: 5, error-to-confidence: 4, confidence-to-reflexion: 3, inhibition-to-reflexion: -6.

- Short-term: 0 for all, except for the auto-excitatory short-term of the rule-coding cluster randomly activated at the begining. Short-term that will not be update during the test are set to 1.

- Thresholds: memory and intention: 3, output: 4, rule-coding clusters: 2, error: 5.5, go: 3, reflexion: 5, inhibition: 4.

For neurons activation:

$$s_i(t + 1) = F\left[\sum_j W_{ij}(t)s_j(t) - T_i + N\right] \tag{1}$$

where $s_i(t)$ is the i cluster activity at t time, $W_{ij}(t)$ the synaptic efficacy from cluster j to cluster i, T_i the activation threshold of cluster i and N is a noise term with uniform distribution whose range is between -0.7 and 0.7 in the paper, but that has been decreased to -0.5 to 0.5 to avoid some memory competition problems.
F(x) function is a sigmoid:
$$F(x) = 1/\left(1 + e^{-x}\right) \tag{2}$$

Synaptic efficacy $W_{ij}(t)$ is computed as follows:
$W_{ij}(t) = S_{ij}(t)L_{ij}(t)$
With $S_{ij}(t)$ the short-term component representing other cluster influences on the synapse and $L_{ij}(t)$ the long-term component.

Short-term updates
The short-term components that will be updated during the test, are set to 0 at the beginning. The influence of the cluster m on a synapse between cluster i and j is mesured at each time step by the following formula:

$$S_{ij}(t+1) = \begin{cases} \alpha S_{ij}(t) + 1 - \alpha, & \text{if } s_m(t) > 0.5 \\ \alpha S_{ij}(t), & \text{if } s_m(t) < 0.5 \end{cases} \tag{3}$$

With $\alpha = 0.4$. The connections concerned are the following ones: input-to-memory, memory-to-intention, intention-to-output, output-to-output and inhibition-to-inhibition. For the intention-to-output short-term updates, another thing has been added, the connection are decreased if go activity is below 0.5 or if there is some hesitation about the intention, i.e if at least 2 intentions have an activity above 0.02 to avoid go activity to trigger a motor output when memory is changing and so intentions are changing. This 0.02 value is rather extreme but by analysing intention activities and testing different thresholds, it is the best and the most accurate value possible.

The auto-excitatory connections update of the rule-coding clusters are made accordingly to error and rule activities: :

$$S_{ii}(t+1) = [\sigma S_{ii}(t) + 1 - \sigma]\,[1 - Q(t)]$$
$$+ \delta S_{ii}(t)Q(t)$$

(4)

with σ the recovery rate of a synapse (equal to 0.99 for a long memory of rejected rules) and $\delta = 0.97$. With

$$Q(t) = [s_r(t)s_i(t)]^2$$

(5)

, where $s_r(t)$ is error cluster activity, and $s_i(t)$ the one of rule-coding cluster i.

Finally, the last short-term components to be updated are the ones connecting intention to error layer:

$$S_{ir}(t+1) = \begin{cases} \delta S_{ir}(t) + 1 - \delta, & \text{if } s_r(t) > 0.5 \text{ and } s_i(t) > 0.5 \\ \delta S_{ir}(t), & \text{else} \end{cases}$$

(6)

All the other short-term components are set to 1 during all the test.

Long-term components update
Only memory-to-intention long-term components are updated in the test, using a learning Hebbian rule linked to error:

$$L_{ij}(t+1) = L_{ij}(t) - \beta s_r(t)S_{ij}(t)s_j(t)$$
$$\cdot [2s_i(t) - 1]$$

(7)

5.2 Other activities graphs.

Here is an activities graph of a complete WCST test, showing the behaviour of the model, trying different rules when an error is made. We can see that the rule activities fit rather well to the hidden rules to be found.
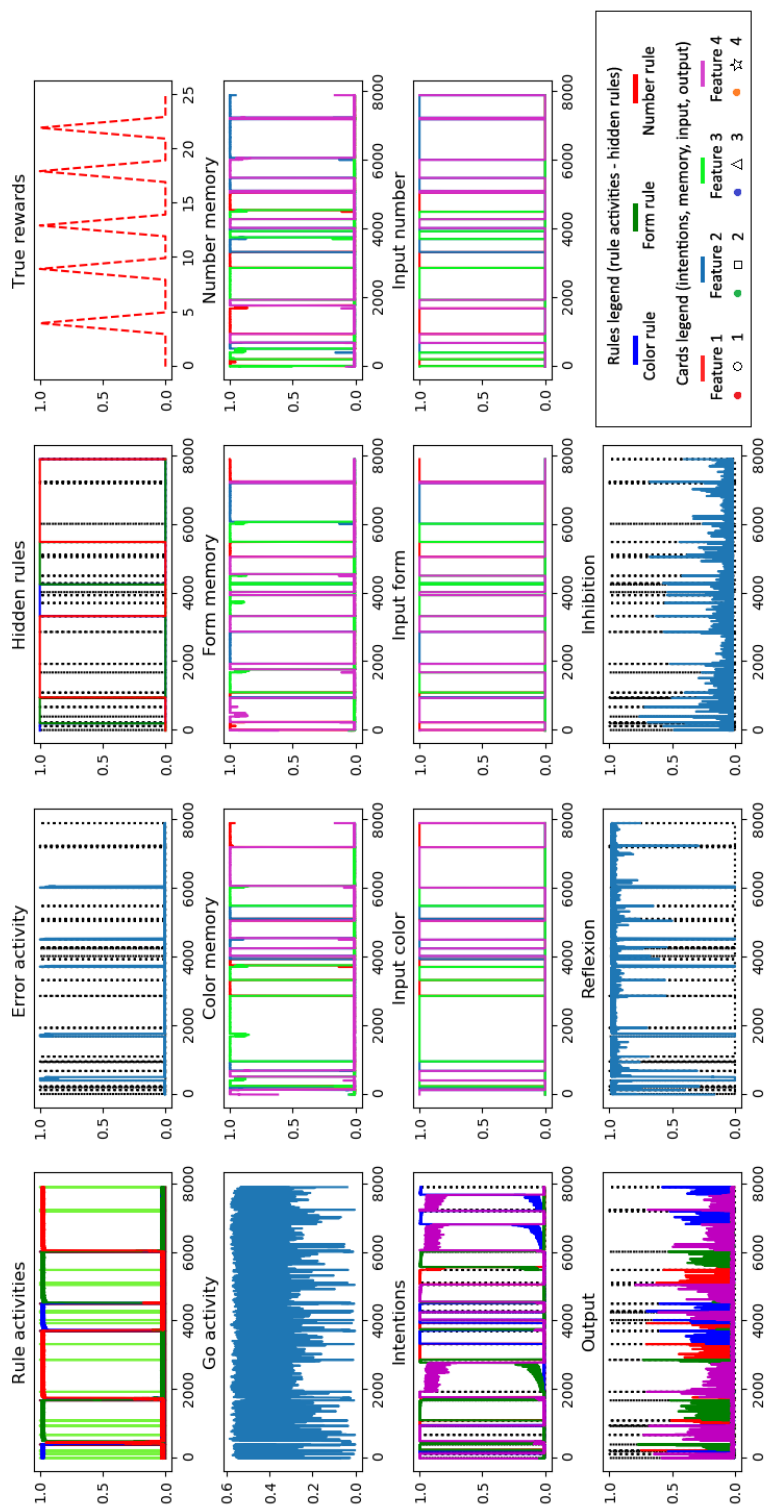
Figure 6. Recording of clusters activities made with matplotlib library and pyplot module. *The card legend colors represents each reference cards (card 1 red, card 2 blue, card 3 green and card 4 violet). In rule activities, a light green dash represents a successful trial and a dark red dash, a negative trial. In other graphs, the trials are represented by dotted-black-dash.*

References

1. R. K. Heaton. "Wisconsin card sorting test manual." In: **Psychological Assessment Resources** (1981).
2. S. Dehaene and J.-P. Changeux. "The Wisconsin Card Sorting Test: Theoretical analysis and modeling in a neuronal network." In: **Cerebral Cortex** 1.1 (1991), pp. 62–79.