Extracting and exposing predictive cortical columns for selective attention

David Eriksson

Independent researcher

Sweden

Abstract

We believe that the degree of importance of a particular stimulus is based on statistical and informational aspects of the environment rather than implicit processing constraints of the observer. Therefore we have designed an algorithm that finds important stimulus based on pairs of input and output for different tasks. An important stimulus shows first order correlation with the output for the particular task. The problem is thus to separate the "correlations" for the different tasks. The algorithm is tested with two situations, one in which the task balance sensitivity is evaluated and another that tests a more complex setup.

Introduction

The motivation for the algorithm presented in this paper is that the degree of importance of a particular stimulus is based on statistical and informational aspects of the environment rather than implicit processing constraints of the observer [1]. For example some stimulus may be important in order to handle particular task. However, for another task the same stimulus may not bring any information and may even contribute with conflicting information. The term that we will use is that the stimulus mentioned above is *reliable* for task A but *un-reliable* for task B [1]. In particular in this paper we assume that a reliable stimulus must show a first order correlation with the correct handling (action) of that task. In this paper we will also call a unit reliable if it represents a reliable stimulus.

The problem of finding reliable stimulus for continuous stimulus has been addressed in terms of variance based competitive combination of inputs [1]. In this paper we will present an algorithm that finds the reliable stimulus for binary stimulus and for multiple task situations.

The algorithm could be interpreted as a row of units to which stimuli (input) terminate from below and the correct action (cross)-terminates from above (see figure 1).

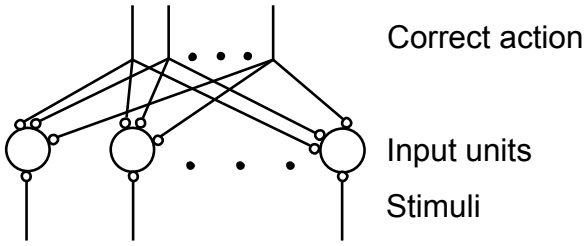

Figure 1. Neural interpretation of the algorithm.

In order to explain the function one could divide the algorithm into two phases. One in which the correct action and stimuli is associated, in terms of adjusting the weights for different tasks and stimuli. These weights will then be used in the next step by the correct action to generate the input [2]. This generated input is then compared to the stimuli in order to find deviations. The comparison is based on the unit-wise sum of generated input and input (stimuli) and not based on the unit-wise difference. Since the input activity is binary a mismatch will result in a more average sum, whereas if there is a match the sum will be more extreme.

Method

To test the algorithm we have chosen the BCPNN-model since it has few parameters [5]. Weights are normalized with the mean activity of the pre- and post synaptic unit so weights are less sensitive to the presentation frequency of stimuli, etc (See equations below). The BCPNN-model is a rate model and is inspired by the groups of interdependent neurons seen in visual cortex, i.e. the orientation selective cells. As

such the model treats a single stimulus as an interval code among a group of neurons (stimuli and correct action must be composed of groups).

The bias for each unit has been omitted since the output activity should only be dependent on the synaptic input and not on the unit's mean activity. Correct action, input units and stimuli are denoted as u, v and I. Indices (i) denote which module and (i') denote which unit inside the module. $\alpha_{ii'}$ is the input gain.

$$h_{ii'}^0 = 0, v_{ii'}^0 = 0$$
$$P_{ii'}^0 = 0, P_{ii'}^0 = 0, P_{ii'jj'}^0 = 0$$
$$P_{ii'jj'}^n = P_{ii'jj'}^{n-1} + \tau_L^{-1}((1-\lambda_0{}^2)v_{ii'}^{n-1}u_{jj'}^{n-1} + \lambda_0{}^2 - P_{ii'jj'}^{n-1})$$
$$P_{ii'}^n = P_{ii'}^{n-1} + \tau_L^{-1}((1-\lambda_0)v_{ii'}^{n-1} + \lambda_0 - P_{ii'}^{n-1})$$
$$P_{jj'}^n = P_{jj'}^{n-1} + \tau_L^{-1}((1-\lambda_0)u_{jj'}^{n-1} + \lambda_0 - P_{jj'}^{n-1})$$
$$w_{ii'jj'} = \frac{P_{ii'jj'}^n}{P_{ii'}^n P_{jj'}^n}$$
$$h_{ii'}^n = h_{ii'}^{n-1} + \tau_R^{-1}\left(\left[\sum_j^H \log\left(\sum_{j'}^{U_i} w_{ii'jj'}u_{jj'}^{n-1}\right)\right] + \alpha_{ii'}\log(I_{ii'}) - h_{ii'}^{n-1}\right)$$
$$v_{ii'}^n = \frac{e^{y_{ii'}^n}}{\sum_{k\in U_i}^{U_i} e^{y_{ik}^n}}$$

Since the proposed algorithm compares the input and generated input we need to define this comparison or difference for the interval-coded stimulus. If we assume that the input is binary (one of the units in the group is 1 and the others are 0), we can, for each unit in the group, add the generated input and input and then measure the difference between them in terms of the entropy among the neurons in the group. Additionally the generated input must sum to one inside the group, i.e. it must match the sum of the input. For example if we have a group with two units. The binary input is [0,1] and the generated input is [alpha, 1-alpha]. After unit-wise summation we get [alpha, 2-alpha]. The entropy will be maximized (1 bit)/minimized (0-bit) as the difference between input and generated input is maximized/minimized.

The contrast evaluating function of the entropy calculation among a group of units can be approximated by a mirroring sister group with subtractive normalisation. For example, if the group has activity [1,0] or [0.5, 0.5] the activity in the sister group after convergence will be proportional to [1,0] or [<0.5, <0.5]. The sum of the activity inside the group corresponds to the contrast.

In order to adapt to changes in correlation between stimuli and correct action, the strength of the stimuli must be larger than the generated input, i.e. the stimuli must dominate the activity in the input unit. If that wouldn't be the case the generated input will "feed" itself. This is because the dominating generated input will bring the weights further to the generated input's favour and as a result producing even stronger generated input.

To test the algorithm we present stimuli and correct action pattern for different tasks. For each task there can be more than one stimuli condition and correspondingly more than one correct action. Such a stimuli pattern is defined by means of the reliable groups. For example, if there are four different stimuli conditions and two reliable groups of two units each, these four units could be [01,01], [10,01], [01,10], [10,10]. The rest of the pattern should consist of randomised activity with one active unit per group. To reflect unreliability, the unreliable part of the pattern should be randomised such that two consecutive stimuli patterns differ, even if the reliable parts are equal. To each such sub-pattern (corresponding to the reliable part) there should be a corresponding correct action pattern. Choosing the patterns like this gives rise to a correlation between the reliable part of the stimuli pattern and the correct action pattern.

In order to automatically create different versions of this test the reliable part of the stimuli pattern as well as the correct action pattern can be randomly generated (One active unit per group).

Each weight will correspond to the mean correlation between the corresponding stimulus unit and correct action unit for all tasks. Thus there can emerge two kinds of deviation between stimuli and generated input: inter task deviation and intra task deviation. Intra task deviation is when the generated input is based on correlations that are especially strong for the same task as during which the deviation emerges. On the other hand inter task deviation is when the generated input is based on correlations that are especially strong

for a different task than the one during which the deviation emerged. It this later type of deviation that we will focus on since it can be used to distinguish correlations between the different tasks. For both types of deviation it is true that a large (small) deviation corresponds to an unreliable (reliable) stimulus.

The algorithm will be tested with two situations. In the first situation the algorithm is presented with a task (get a banana or to signal), a stick (which can be long or short and bright or dull) and the correct action ("will work" or "will not work"). A conflict in this situation is that a bright stick will work when the task is to signal, whereas a bright stick will not work when the stick is short and the task is to reach a banana. This means that there are eight correct combinations:

- ***Task**: Banana. **State-pair**: (first-state: long and bright, following-state: "will work").*
- ***Task**: Banana. **State-pair**: (first-state: short and bright, following-state: "will not work").*
- ***Task**: Banana. **State-pair**: (first-state: long and dull, following-state: "will work").*
- ***Task**: Banana. **State-pair**: (first-state: short and dull, following-state: "will not work").*
- ***Task**: Signal. **State-pair**: (first-state: long and bright, following-state: "will work").*
- ***Task**: Signal. **State-pair**: (first-state: short and bright, following-state: "will work").*
- ***Task**: Signal. **State-pair**: (first-state: long and dull, following-state: "will not work").*
- ***Task**: Signal. **State-pair**: (first-state: short and dull, following-state: "will not work").*

All these combinations are shown to the algorithm. The difficulty in the experiment is then to expose the reliable module for each task (Banana => Length of the stick and Signal => Brightness of the stick). In order to make the experiment more realistic the two tasks (banana (A), signal (B)) and the different properties of the stick (long (A1), short (A2), bright (B1), dull (B2)) are presented according to the probabilities: P(A)=S, P(B)=1-S, P(A1) = A, P(A2) = 1-A, P(B1) = B and P(B2) = 1-B. As a result the output becomes: P(1) = R = S+(1-S)B and P(2) =1-R= S(1-A) + (1-S)(1-B).

The performance of this simple test has been measured in terms of the difference in the entropy between the reliable (task dependent) and the unreliable group.

During 100 iterations all the eight correct combinations are fed to the network, which yields about 12 iterations per combination if S=0.5, A=0.5 and B=0.5. More precisely each combination will run for

$100*(S)^{banana}*(1-S)^{signal}*(A)^{long}*(1-A)^{short}*(B)^{bright}*(1-B)^{dull}$ iterations.

These 100 iterations are then repeated three times. The parameters used in this experiment are: τ_L=500, λ_0=0.0001, τ_R=1 and $\alpha_{ii'}$=0.1. The difference in the entropy is measured during the last 100 iterations.

The more complex test has five different tasks. Each task can be handled in four different ways, i.e. there are four different stimuli-action associations for each task. There are 10 groups of input and output units and each group contains four units. For each task two of these 10 input groups are reliable. In one setting these two reliable groups are chosen randomly. In another setting they are non-overlapping between the tasks, i.e. the first two groups are reliable for the first task and the second two groups for the next task etc.

Each stimuli-correct action pattern pair is clamped for three iterations. For each of the four ways that a task can be handled, 10 different randomised versions of the stimuli pattern (with constant reliable part) will be presented. A set of five tasks will be repeated 10 times. The entropy is measured during the last repeat. To conclude there will be 3*10*4*5*10 iterations.

The performance of this more complex test has been measured in terms of weather, or more exactly how many groups of the pair of groups with the lowest entropy coincide with the reliable groups.

For a whole test (five tasks) the performance is measured as the number of coinciding groups divided by ten (the total number of groups). This ratio is averaged over 30 (randomised) versions of the test.

The parameters used in this experiment are: τ_L=500, λ_0=0.0001, τ_R=1 and $\alpha_{ii'}$=1.

Result

Below we have plotted (figure 2) the correctness of the first test. We vary the balance of the two tasks (Only "signalling"-combinations are presented for S=0 and an equal number of "signalling" and "get banana" for S=0.5) and the balance between the properties of the stick (A=B, so when A=B=0, then only the short and dull stick-combinations are presented to the algorithm).

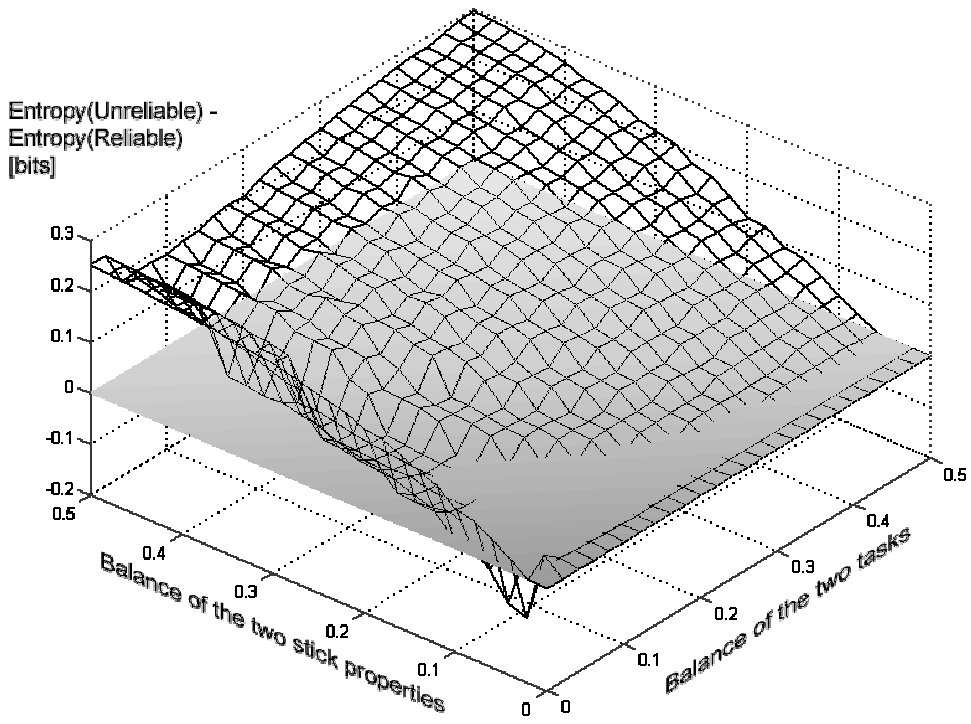

Figure 2. Difference between the entropy in the unreliable group and the reliable group.

In the figure we can se that the maximum difference is 0.3 bits between the un-reliable and reliable module. The maximum occurs when all combinations are presented for an equal period of time, i.e. when both the stick property balance and the task balance are 0.5. The difference is less sensitive to the balance of the two tasks than the balance of the two properties of the stick. The algorithm exposes the un-reliable module more than the reliable when the stick property balance is less than 0.075.

We are not certain about the origin of the "step" in the figure at low task balance, but we believe that it is due to the normalisation with the task count, i.e. the task specific contribution to the total entropy difference is down-weighted with the task count.

The performance for the more complex test for randomised (overlapping) reliable groups is 0.76 and for no overlapping reliable groups it is 0.98. However it is important to note that although the lower performance suggests that the algorithm can't handle overlapping cases, simple, unpublished tests with overlapping reliable groups show that this conclusion is wrong.

Discussion

We have now presented an algorithm that can find the reliable stimuli for different tasks. Although the reliable stimuli are found in terms of a group of units it would be interesting to see if the principle can be used to find individual reliable units.

Moreover we believe that it is off extra interest to find the reliable groups of interdependent units for two reasons. First, groups of interdependent units could exist in primary sensory areas [3,4 and 6]. Second, if one unit among a group of interdependent units is reliable for a particular task, the dependency suggests that the other dependent units could be reliable as well. Therefore organising interdependent units into groups imposes a natural spatial constraint for attention and assessment of plasticity.

Acknowledgements

We thank Erik Fransèn for his helpful comments on early versions of the manuscript. We would also like to thank Margareta (the author's mother), Sten (the author's father) and Daniel (the author's older brother) for making this work possible in a number of ways.

References

[1] Dayan, P, Kakade, S and Montague PR. Learning and selective attention. Nature, 3 (2000) 1218-1223.

[2] Hinton, GE, Dayan, P, Frey, BJ and Neal, RM. The wake-sleep algorithm for unsupervised neural networks. Science, 268 (1995) 1158-1160.

[3] Roland, PE and Zilles, K. Structural divisions and functional fields in the human cerebral cortex. Brain Research Reviews, 26 (1998) 87-105.

[4] Rolls, ET and Treves, A. Neural Networks and Brain Function. (Oxford University Press, 1998).

[5] Sandberg, A, Lansner, A, Peterson, KM and Ekeberg, Ö. A Palimpsest Memory based on an Incremental Bayesian Learning Rule. Neurocomputing, 32-33 (1999) 987-994.

[6] Tanaka, K. Zeroing in on the higher brain functions. Riken Research Highlights News. Brain Science Institute. Laboratory for cognitive brain mapping. 251 (2002).