Cortical Columns as the Base Unit for Hierarchical and Scalable Machine Learning

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Studying the cortex via columnar organization of neurons is a perspective. One which is more holistic than single cell models but still far less ambitious than studying diverse networks [1]. Though there are confusions in terminology, studying columnar organization instead of neurons have allowed many researchers to attain a functional understanding of and ability to model processes such as map cell placement [9] and pattern recognition. We report an extension to a model proposed by A.G. Hashmi, M.H. Lipasti. We discuss the biological plausability of cortical column modeling and show a two level implementation.

1 Biological Background

Lorente de No first described what are now known as cortical columns in 1938 [7]. Then, he suggested that these originations of cells could be called an "elementary unit" which builds to brain function. In 1957, Mountcastle gave the name cortical columns. Since the initial descriptions and definitions, the term "cortical column" is widely used and is the source of some confusion. In addition, the term hyper column is used as well. Here, we will refrain from using "general" terms as possible.

A "cortical column" in the broad sense, refers to the vertical organization of neurons found in the cortex. We use the terms minicolumn and macrocolumn to be more precise. A minicolumn is a vertically organized group of neurons, between 80-100, which are responsible for independent receptive fields. These minicolumns communicate with other minicolumns in a organization termed the macrocolumn. Each macrocolumn has found to contain between 60-80 minicolumns [1].

Much of the interest in cortical columns arise from how they appear to code for information. The minicolumns of each macrocolumn appear to traverse laminas II - VI with feed-forward, feed-back, and horizontal connections. Calvin (1998) has suggested that laminas II and III code for associations, laminas V and VI code out, and lamina IV receives the inputs [6]. These mechanisms also seem to represent one orthogonal type of input, where minicolumns use lateral inhibition to sharpen their borders and increase their own field's definition [5]. The mechanism for this process has been suggested to be derived from axon bundles of double bouquet cells [5].

These attributes are even more attractive because no research has yet to find minicolumn activity outside its own macrocolumn. This suggests that they act as independent feature detection and processing agents. In-vivo mouse studies show that in the barrel cortex the minicolumn excitation and inhibition connections work to enhance important information while suppressing distracting information [4].

Furthermore, these general methodologies for processing are nearly identical throughout the central nervous system. Columns integrate a diverse set of information; including auditory, visual, sensory, motor, and memory [1]. So far, the studies of these species support the concept of using cortical columns as functional units for brain processing. However, some refute the use of cortical columns because of their limited appearance in many species with similar visual function [8]. Nonetheless, at least in applicable species cortical columns remain very attractive for conceptual and modeling studies.

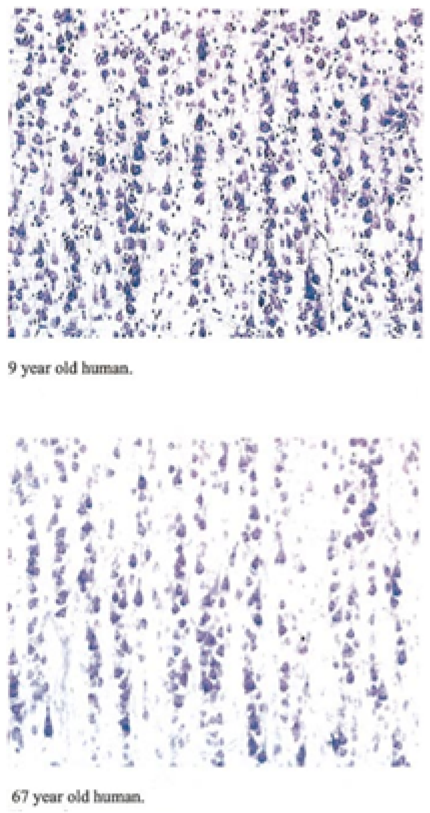

Fig. 1 Minicolumns according to cell soma arrays in planum temporale at various ages. Lamina III, 35-μm thick Nissl-stained slides, celloidin embedded. Yakovlev–Haleem collection. 100× total magnification.

Figure 1: Adapated from [1].

2 The Predictor Module

Predictor modules were used to model each cortical column[10]. Each module consists of two units: a predictor unit and a code unit, and exhibits both vertical and horizontal connections. Each module will code for an independent feature by performing predictability minimization of its code unit.

2.1 Requirements

This model, based on predictability minimization, was chosen for multiple reasons.

Biological Plausibility Though based on the perceptron, the model described here resembles biological cortical columns when considering a higher level of abstraction (i.e., ignoring individual neurons). The horizontal connections found between predictor unit of one module and the code units of the others mimic connections found in layers II and III between multiple cortical columns. The code unit, on the other hand, resembles Purkinje cells in the cerebellum: neurons will modify their synaptic weightings when receiving simultaneous activations at two of its inputs so that this does not occur again.

Scalability To increase the amount of features for which the system encodes, we simply add more predictor modules. The amount of operations executed during the training process increases linearly with the amount of predictor modules. Dimensionality of the input data is similarly not a problem.

Hierarchical Though single-layer systems are considered for this explanation, the idea can be extended. The outputs of a series of modules can give rise to the inputs to another layer. These modules can therefore be "stacked" on top of each other.

2.2 System Dynamics

As mentioned before, each predictor module is made up of one predictor unit and one code unit. Code units will lock on to different features while their predictor unit will assure the independence of the coded feature. To do so, an `objective function` is considered (such as an error function) which is maximized by the code unit and minimized by the predictor unit by use of backpropagation; a coevolution of both can then occur until an equilibrium is reached.

3 Methods & Implementation

The model has been implemented in MATLAB®. Multiple systems were tested (single- and multi-layer) with datasets of varying size and dimensionality. The notation used is explained below.

Symbol	Explanation
x^p	The p^{th} input vector.
y_i^p	Output of the i^{th} code unit, given input p, x^p.
P_i^p	Output of the i^{th} predictor unit, given input p, x^p.
$\bar{y}_i$	Average output of the i^{th} code unit.
z^p	Reconstruction of input pattern p, x^p.
U	Code unit weights.
V	Predictor unit weights.

3.1 Objective function

Initially, the following error function was considered:

$$T = \alpha V - \beta I - \gamma H \tag{1}$$

Where $\alpha, \beta, \gamma > 0$ are constants, $V = \sum_{i,p}(\bar{y}_i - y_i^p)^2$, $H = \sum_{i,p}(P_i^p - \bar{y}_i)^2$ and $I = \sum_p (z^p - x^p)^T(z^p - x^p)$. However, this can be simplified as explained by J. Schmidhuber **??** to:

$$V_C = \sum_{i,p}(P_i^p - y_i^p)^2 \tag{2}$$

3.2 Code units

Once the input patterns are fed into the code units, their output is calculated and compared to the output of their corresponding predictor units. Their weights are updated by gradient ascent of the of the error function. A scaling factor inversely proportional to the absolute values of the current weights is introduced to prevent much variation once the module has locked on to an independent feature.

$$\hat{U}_i = U_i + \delta U_i = U_i + \alpha \frac{\partial E_i}{\partial U_i} = U_i - \alpha \times (P_i - \sigma(y_i))(1 - \sigma(y_i))\sigma(y_i) \times \eta \times x^p \tag{3}$$

Where $\alpha = (\prod_{j=1}^{m} |U_i^j|)^{-1}$ and η is the learning rate.

3.3 Predictor units

The vector V, containing the weights for the predictor units, must also be updated during each iteration. Predictor unit i receives its input, $\mathbf{y_i}$, which can be described as a vector whose elements, y_i^j, are the outputs of all code units, such that $i \neq j$.

A similar method is then used to update the weightings, this time using gradient descent (since the objective function is to be minimized by the predictor units) as well as a different learning rate.

4 Results

A simple two-level system has been set up, with four cortical columns on the first level and one on the second. Four patterns are presented to the first level, where each cortical column locks on to one (and only one) 'feature' (i.e. pattern). After each iteration of the training process, and output vector is generated when all four patterns are presented, encoded and then fed into the second level of the system. One iteration of the training process is executed on the second level using the newly generated pattern (i.e. recognize the larger pattern, made up of four smaller ones) as a whole.

Once the whole process has been completed, we can see which patterns each cortical column fires for, as can be seen in Figure 2.

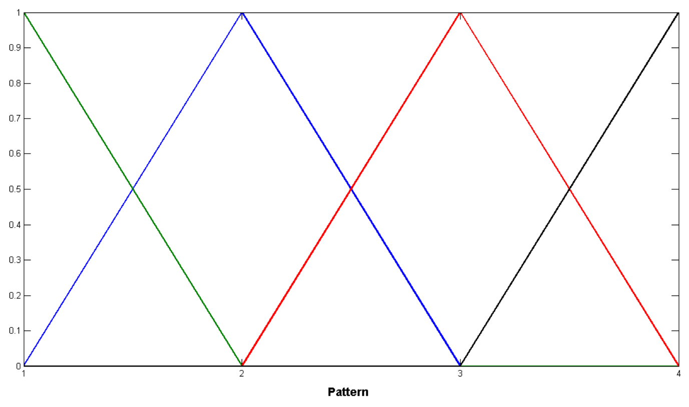

Figure 2: Pattern numbers are found on the x-axis. Each colored line represents the firing state of a different cortical column.

An interesting property exhibited by the system (especially visible when considering the second layer) is that cortical columns may still fire when presented with a similar (yet not identical) pattern for which they encode.

5 Future Work

After verifying that the described system can perform unsupervised learning when presented with an array of different input patterns, as well as constructing multi-layered systems, a number of potential improvements were considered. These improvements are related to both software and hardware.

First and foremost, a nonlinear relationship between learning rates (both for the predictor units and the code units) and dimensionality of the input patterns (as well as the size of the dataset) was observed for accurate pattern recognition. These relationships could be determined experimentally, but was not done due to time constraints. The total number of iterations for each of the two nested loops which are executed for optimal learning conditions can also be determined experimentally, although there will probably exist an interdependence between these values and those of the learning rates. The final software-related improvement is connected to cortical column thresholds. The current system defines an independent threshold for each cortical column after training has finished;

however, it might be interesting to experiment with different setups (one threshold value for all cortical columns, for example).

Once all corrections and improvements have been implemented, it is important to start looking at the hardware side of the system. Current hardware is not designed for machine-learning algorithms. GPGPUs (General Purpose Graphical Processing Units) may provide a better foundation for this method due to the inherent parallelism found on these chips. This implementation would require translation of the algorithm to a lower-level language (such as C), and has already been performed by researchers at the University of Wisconsin [12]. However, initial testing sheds light on some inefficiencies found in these setups.

6 Conclusion

Although not applicable to all species, cortical column modeling is an effective approach to achieve unsupervised learning and information processing. In those species that display cortical columns, especially primates with higher function, the cortical column perspective should be considered. As we show, the models are able to process information and report to other layers. Importantly, this abstraction allows a macrocolumn to report to other regions of the brain, i.e. the other hemisphere or hypothalamus, with one connection. The independence of macrocolumns prunes the number of connections which report to other areas of the brain, which is not directly achievable or biologically plausible in neuron networks [1].

References

[1] D. Buxhoeveden, M. Casanova. The minicolumn hypothesis in neuroscience. Brain 125(2002):935-951.

[2] V. Mountcastle. The columnar organization of the neocortex. Brain 120(1997):701-722.

[3] D. Hubel, T. Wiesel. Functional Architecture of Macaque Monkey Visual-Cortex. Proceedings of the Royal Society B: Biological Sciences 198.1130 (1977):1-.

[4] J. Porter, C. Johnson, A. Agmon. Diverse types of interneurons generate thalamus-evoked feedforward inhibition in the mouse barrel cortex. J Neurosci 2001; 21: 2699-710.

[5] J. DeFelipe, I. Farinas. Chandelier cells and epilepsy. [Review]. Brain 1999; 122: 1807-22.

[6] W. Calvin. Competing for consciousness: how subconscious thoughts cook on the back burner. [Online]. 1998, April 30; Available from: http://williamcalvin.com/1990s/1998JConscStudies.htm last updated April 30, 2001.

[7] Lorente de No R. The cerebral cortex: architecutre, intracortical connections, motor projections. In: Fulton JF, editor. Physiology of the nervous system. London: Oxford University Press; 1938. p. 274-301.

[8] J. Horton, D. Adams. (2005). The cortical column: a structure without a function. Philosophical Transactions of the Royal Society B: Biological Sciences, 360(1456), 837-862.

[9] Martinet et al. Map-Based Spatial Navigation: A Cortical Column Moel for Action Planning 2008

[10] J. Schmidhuber. Learning Factorial Codes by Predictability Minimization. University of Colorado, 1991

[11] A.G. Hashmi, M.H. Lipasti. Cortical Columns: Building Blocks for Intelligent Systems. University of Wisconsin, 2009

[12] A. Nere, M. Lipasti. Cortical Architectures on a GPGPU. University of Wisconsin, 2010.