A Bayesian Model for Canonical Circuits in the Neocortex for Parallelized and Incremental Learning of Symbol Representations

Martin Dimkovski and Aijun An

Technical Report CSE-2013-10

October 1 2013

Department of Computer Science and Engineering
4700 Keele Street, Toronto, Ontario M3J 1P3 Canada

A Bayesian Model for Canonical Circuits in the Neocortex for Parallelized and Incremental Learning of Symbol Representations

Martin Dimkovski*, Aijun An

York University, 4700 Keele St. Toronto, ON M3J 1P3, Canada

Abstract

We present a Bayesian model for parallelized canonical circuits in the neocortex, which can partition a cognitive context into orthogonal symbol representations. The model is capable of learning from infinite sensory streams, updating with every new instance and without having to keep instances older than last seen per symbol. The inherently incremental and parallel qualities of the model allow it to scale to any number of symbols as they appear in the sensory stream, and to transparently follow non-stationary distributions for existing symbols. These qualities are made possible in part by a novel Bayesian inference method which can run Metropolis-Hastings incrementally on a data stream, and significantly outperforms particle filters in a Bayesian neural network application.

Keywords: neocortical canonical circuits, Bayesian brain, symbolic abstraction, incremental Metropolis-Hastings, data stream learning

1. Introduction

We present a model for a hypothetical functional unit of the neocortex and its relationship with proximal peers which share the same cognitive context. Our model is not one of the neocortex at large, which would require a network of cognitive contexts, but we hope it offers a building block for such

*Corresponding author. Tel: +1-416-452-8071

Email addresses: martin@cse.yorku.ca (Martin Dimkovski), aan@cse.yorku.ca (Aijun An)

an objective. Our approach is to first identify key computational aspects of the neocortex, and then build the model upon those assumptions. We demonstrate how the resulting model can orthogonalize a cognitive context developing representations for the cognitive symbols. The model is evaluated on popular machine learning datasets.

Our assumptions are detailed in section 2. The principal assumption is the existence of an elementary functional unit in the neocortex, identified as a canonical circuit. Next, we assume that each canonical circuit develops to represent a particular cognitive symbol, by learning towards data associated with its symbol, and learning away from the data of all other symbols. Another assumption is that each canonical circuit must execute concurrently with all other canonical circuits, in complete task parallelism. Next we assume that neocortical computation is analogous to Bayesian inference, and we approach this aspect through Marr's three levels of analysis. Lastly, we assume that canonical circuits must operate inherently incremental by learning with few examples, from infinite data streams, and without having to store old data or use multiple epochs. There is existing work in neocortical computational modeling which covers the previous listed assumptions to a certain extent either individually or in subsets. However, we are not aware of any work which covers all the assumptions jointly. One of our contributions is that we identify the state of each aspect in current neuroscience, propose correlations between them, and propose a model that puts them all in a common framework.

In our model, each cognitive symbol is represented by a canonical circuit in the form of an independent Bayesian neural network. Each of these neural networks updates with its own Bayesian inference process, yet coupled inhibitory to those of other symbols, so that each pursues uniqueness and the overall result is orthogonalization of the cognitive context which describes the data stream. The model starts blank and adds a canonical circuit for each new symbol as it shows up in the stream. For example, cognitive context could be "direction of motion" with its symbols being "up", "down", "right", etc. Figure 1 shows a simplified visualization of how canonical circuits orthogonalize a cognitive context. Due to the task parallelism, regardless of how many canonical circuits become involved, the model runs in constant time, and the canonical circuits can be distributed to different processors or machines across the network.

In order to meet the requirements for incremental learning, we developed a novel Bayesian inference method which runs Metropolis-Hastings (MH) on

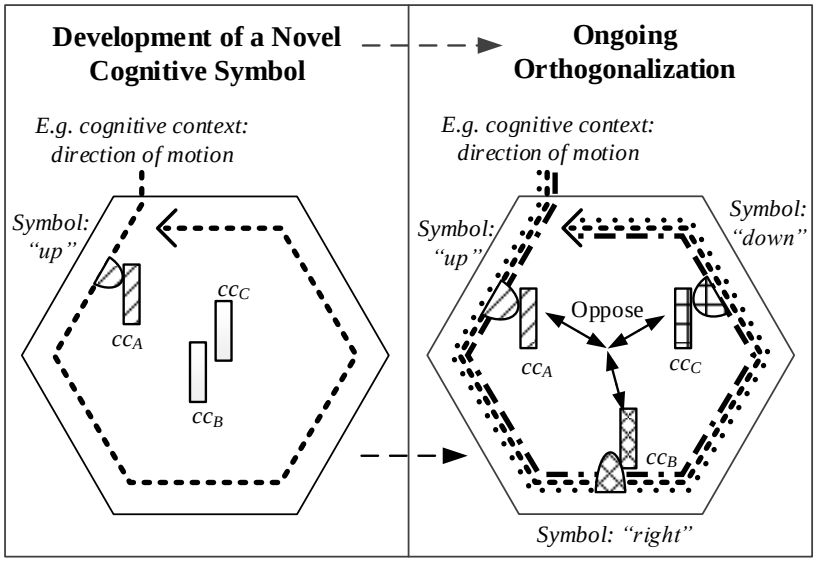

Figure 1: Example orthogonalization of a cognitive context. The left plate shows the development of a canonical circuit cc_A for the first symbol that shows up in the streaming sensory input. The other canonical circuits are not initialized yet. The right plate shows the state after each canonical circuit has associated with a symbol. From here on, they all continually specialize for their symbols while opposing each other.

a data stream. The method makes it possible for a single data instance to be sufficient in forming a useful representation of a symbol, and for each symbol to update with each new data instance efficiently. No data instances, before the last seen per symbol, have to be kept for subsequent updates. We discuss how it is possible in an optimal model for not even the latest instance to be required. We call our method Incremental Metropolis-Hastings (IMH). IMH recurrently re-uses the last posterior as a new prior. Priors and posteriors are represented as non-parametric probability distributions, utilized through Monte Carlo or kernel density estimators. Therefore the inference does not suffer from limitations of point-based approximation such as Maximum-a-Posteriori.

From a purely computational perspective, we contribute a Bayesian classification model which is capable of supervised learning of unlimited number of symbols (classes) from an infinite data stream, and which has simple pa-

rameterization. Because of its incremental learning qualities, the model is unique in handling concept drift transparently, i.e. it inherently supports non-stationary class distributions. We show that it matches the performance of state-of-the-art incremental learning methods. IMH is also a computational contribution in itself, because we show it vastly outperforms particle filters for incremental Bayesian inference, at least with a neural network model.

In the following section we present the background for our model, structured as a literature review of the principal assumptions, each of them identified as a subsection. We attempt to relate them by expressing them into a shared terminology, which allows for a unified perspective upon which we build the model. In the third section we describes our model in details. The forth section reviews existing and related models. In the fifth section we present the evaluation results, after which we finish with a Conclusions section.

2. Background

2.1. Canonical circuits in the neocortex

The idea of elementary circuits as functional modules in the neocortex was hypothesized as early as 1938 [1], though it remains an open question [2]. A prominent hypothesis of this type is the columnar view of the neocortex, based on functional identification of neural circuits perpendicular to the pial surface [3, 4, 5], as well as a repeating template of neural distribution and connectivity found in such circuits [6, 7, 8]. In the columnar hypothesis, the smallest circuit is called a mini-column, and proximally connected mini-columns form a column. Depending on the area of neocortex, a mini-column is linked to a specific representations such as line orientation, isofrequency tones, angles, or direction of motion. Inside a column, its mini-columns typically cover the full range of their representation type, such as full 180° covered for orientation or direction of movement, or the full tone frequency spectrum [3]. Columns are proximally or distally interconnected with many peer columns across the neocortex.

Even though the direct and indirect legacy of the columnar hypothesis is undeniable [9], it hinges too much on anatomical modularity, which is debatable [2, 10]. There is a general consensus however that there is some functional differentiation between neocortical circuits [9]. da Costa et al. [9]

review the legacy of the columnar hypothesis, its criticism, and recent perspectives, and then propose that we focus on the most salient aspects of the idea by using the term *canonical circuits* instead of mini-columns, and not trying to constrain these circuits into rigid physical columns. Their canonical circuit "embodies the idea of a repeated local circuit that performs some fundamental computations that are common to all areas of the neocortex.". In their view, the exact physical location and configuration of canonical circuits can change and even be overlapping [9].

2.2. Symbols and Cognitive Contexts

The canonical circuit has a multifaceted input [2, 7]. One part of the input is from the environment, coming from subcortical structures such as the thalamus. Another part of the input comes from peer cortical circuits which could be anywhere in the neocortex. The environmental and the peer inputs come into the canonical circuit at different locations, and can therefore be seen as contextual to one another, through the canonical circuit as a context processing element. Therefore, we will refer to the combined input as *cognitive context*.

Certain facts put forward in the columnar hypothesis suggest that a column, and therefore all its mini-columns, share a similar set of inputs [5]. The width of a column is also linked to the termination width of the sensory (thalamic) input [4]. In other words, any particular cognitive context could be seen as being processed by a group of canonical circuits. Each canonical circuit in the group represents a particular aspect of the cognitive context, for example a particular angle in the cognitive context "line orientation." We will refer to any particular representation of a cognitive context as a cognitive symbol. Any such symbol has meaning only given its context. The term cognitive symbol has been defined previously, in a compatible perspective, as the internal categorical representation of an external physical or information entity [11]. Identifying the symbol representation mechanism in neural circuits is a critical challenge [12].

2.3. Requirements for parallelism

The assumption of canonical circuits implies that each circuit is distinct from others and can execute in parallel [13]. Functional magnetic resonance imaging during various behaviour shows that multiple distal areas of the neocortex appear as active at the same time. It is therefore unlikely that canonical circuits involved in a particular behaviour have close correlations

of internal states. We assume that the learning method inside one canonical circuit must be able to execute without knowing the states in other canonical circuits, i.e. in complete task parallelism. In other words, each canonical circuit should be able to run on a separate machine.

2.4. Why Bayesian and how?

Bayesian probabilistic inference is becoming a leading candidate for explaining the operation of the brain [14, 7, 12, 15]. Bayesian probability is an evidential probabilistic interpretation, where probability is an abstract concept describing a state of knowledge, as opposed to the view of probability as a frequency [16, 17]. Given new relevant data and given a probability model for that data (called the *likelihood*) in terms of some parameters of interest $\boldsymbol{\theta}$, Bayesian inference provides a way to update an assumed probability of the parameters $p(\boldsymbol{\theta})$ (called the *prior* before the update and *posterior* after the update) using a simple formula:

$$p(\boldsymbol{\theta}|data) = \frac{p(\boldsymbol{\theta})p(data|\boldsymbol{\theta})}{p(data)} \tag{1}$$

The challenge in Bayesian inference comes from how the probability models and distributions are framed, learned, and utilized, and how the inference equation (1) is executed. Solutions from Bayesian inference could be either models of the whole posterior distribution, or point-based approximations such as Maximum-a-Posteriori (MAP). Using the whole distribution as a solution gives much more informative solutions because it describes the probability of the parameters at any possible value. This is important when there is uncertainty, as is usually the case, because the probability distribution can account for it, while a singular value is over-confident. This is especially relevant when the goal is to learn from few examples.

Tenenbaum et al. [12] note how Bayesian inference quickly becomes a suspect when we consider how the brain learns and generalizes all too well from sparse, noisy, and ambiguous data: "If the mind goes beyond the data given, another source of information must make up the difference. Some more abstract background knowledge." The authors note how in different disciplines that study the brain this abstract background knowledge is known under different names, such as *constrains* in psychology and linguistics, *inductive bias* in machine learning, and *priors* in statistics.

To consider the idea of a Bayesian brain systematically, we can look at it through the three levels of analysis in Marr's approach to understanding

information processing systems [18]. The names of Marr's levels can be misleading, however the underlying questions are intuitively distinguishable: the top level (called *computational*) considers what the system does and why, the middle level (called *algorithmic* or *representational*) considers what representations are used and how they are built and manipulated, and the bottom layer (called *physical*) considers how the whole system is implemented physically.

At the top Marr's level we have to consider conscious action. Whether Bayesian inference applies at this level is debatable. Critics argue that many conscious processes are notoriously deviant from Bayesian expectations. However there are arguments that this is only so for a single individual where a particular behaviour is just one sample, and that when looked at collectively, at the population distribution level, behaviour is faithfully Bayesian [19]. There is work showing that the brain operates akin to sampling from the population behaviour distributions. For example, Sanborn et al. [20] show that the rational model of categorization in cognitive science works better with Gibbs sampler and particle filters, than with MAP. The representational and physical Marr's level can be described more as unconscious. Existing Bayesian brain theories focus on the representational level, where they attempt to explain how prior and posterior probability distributions are built and maintained on a conceptual level. We present key examples in section 4.1. There are few attempts in literature that attempt to explain the prior and posterior probability distributions on the physical level in terms of neuron and circuit specifics [21, 22]. In general, this issue is considered unresolved [15, 12]. "Uncovering their neural basis is arguably the greatest computational challenge in cognitive neuroscience..." [12].

2.5. Learning with few examples

We consider it unlikely that the brain archives multiple data instances in their original form, for later use in an epoch-like processing. It is more likely that the streaming data, which the brain constantly experiences, is used only at the time of exposure, after which only their contribution remains in the form of adjustments made to neural networks. Under this assumption a faithful neuromorphic algorithm has to learn with few examples at a time. This means the algorithm should start developing useful symbol representations after the first few related instances, as well as use few instances at each update stage.

The quality of learning with few instances has been noted as essential cognitive science. Tenenbaum et al. [12] addresses this topic and discusses how the brain needs only a few examples of a symbol in order to form an understanding. They point out how children can learn to use a new word after just a few examples of it.

3. Proposed Model

3.1. Overview and Definitions

This work presents two independent but complementing contributions. The principal contribution is a model for incremental and Bayesian orthogonalization of a cognitive context, where each resulting cognitive symbol is adopted by a canonical circuit, executing in parallel with all other circuits. The model describes how canonical circuits relate in a cognitive context, and how cognitive symbols develop while inhibiting each other. We refer to this model as Cognitive Context Orthogonalizing Networks (CCON).

In order to run however, each canonical circuit in CCON needs an incremental Bayesian inference method over a non-linear high-dimensional neural network and running over a data stream (i.e. not requiring multiple epochs with archived data). The most fitting existing method is particle filters. However, as discussed later in sections 4.2 and 5, CCON is too complex for particle filters. Therefore we develop the Incremental Metropolis-Hastings (IMH) method. IMH complements CCON, however it can also be used independently in different problems. IMH is the result of pursuing a more biologically plausible incremental inference, one which could be explained better on the physical Marr's level.

An overview of CCON is illustrated in Figure 2. It is composed of multiple canonical circuits (denoted as $cc_1, cc_2, ..., cc_K$), each specializing for a cognitive symbol (i.e. class) of a single cognitive context. A canonical circuit, cc_k where $k = 1, ..., K$, is a construct consisting of an IMH Bayesian process B_k and a multi-layer feed-forward neural network NN_k whose synapses are controlled by B_k. Each canonical circuit cc_k acts as a binary classifier, that outputs the probability for the presence of its associated symbol in the current data stream instance.

The data stream from which the model needs to learn is potentially infinite and defined as $\mathbb{S} =< e_1, e_2, ..., e_\infty >$, where e_i denotes the labeled training example that arrives at the i-th time point. A training example e_i consists of $\boldsymbol{x}_i$ and y_i, which represent respectively the environmental and peer

input parts defined earlier. Specifically, $\boldsymbol{x}_i$ is a vector of sensory attribute values, and it is the same for all cc_k. y_i is the supervision information coming from distal peers outside the cognitive context, suggesting which symbol is related to $\boldsymbol{x}_i$. Because each cc_k is a binary classifier, y_i needs to be a binary class label. Therefore any original multinomial supervision label is converted to multiple binary labels, one for each cc_k. For example, if there are three classes A, B, C and a training example e_i belongs to class B, then the class label y_i of e_i for cc_B will be a 1, while y_i for cc_A and cc_C will be a 0.

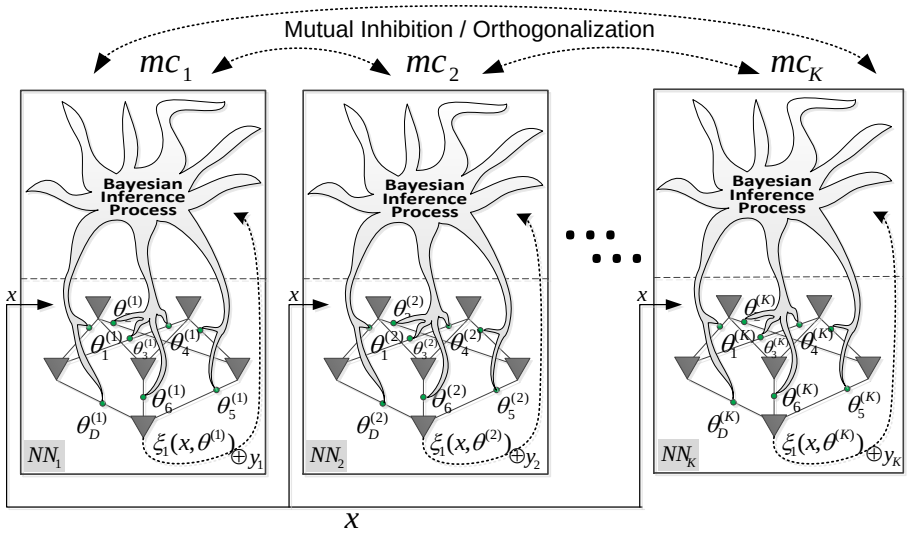

Figure 2: Overview of the CCON model, showing K canonical circuits of a single cognitive context, each circuit with its B_k Bayesian inference process and its NN_k neural network. The same input $\boldsymbol{x}$ feeds all canonical circuits, but each of them has its own supervision information y_i. The top shows that canonical circuits are mutually inhibitive resulting in orthogonalization of the cognitive context.

The neural network NN_k in a canonical circuit cc_k is approximated with a multi-layer feed-forward neural network, fully connected in-between adjacent layers. The input layer is determined as per the dimensionality of input $\boldsymbol{x}$, and the output layer has a single output neuron. Calling upon the universal approximation theorem [23], we use only a single hidden layer. The number of neurons H in each hidden layer can be determined by the user. All weights

$w^{(k)}$ and biases $b^{(k)}$ of NN_k are represented jointly as $\theta^{(k)}$. The output neuron v_{out} uses a log-sigmoid activation function in order to contain the output within 0 and 1. All other neurons use a hyperbolic tangent sigmoid transfer function $tanh$[1]. Thus, given an instance with sensory input $\boldsymbol{x}$, the output of NN_k is:

$$\xi_k(\boldsymbol{x}, \theta^{(k)}) = 1/[1 + exp(-b_{out}^{(k)} - \sum_{h=1}^{H} w_{h2out}^{(k)} v_h(\boldsymbol{x}, \theta^{(k)})] \tag{2}$$

where w_{h2out} is the weight from hidden neuron h to the output neuron, b_{out} is the bias of the output node, and $v_h(\boldsymbol{x}, \theta^{(k)})$ is the output of hidden layer neuron h, computed as:

$$v_h(\boldsymbol{x}, \theta^{(k)}) = tanh(b_h^{(k)} + \sum_{j=1}^{dim(\boldsymbol{x})} w_{inp2h} x_j) \tag{3}$$

The following interpreted execution log provides an idea of how the model works on one of our evaluation datasets (*Wine*). Out of the first 11 instances, only a select few instructive are shown. We can see that CCON starts differentiating early and progressively better. By the 11^{th} instance it is already working with high precision.

Instance 1:
Its symbol: A
First time symbol A is seen, therefore a new cc_A is initialized
How all canonical circuits see this instance at start:
cc_A: Match/Recognition: 0.99999
How all canonical circuits see this instance after learning:
cc_A: Match/Recognition: 0.99999
...
Instance 3:
Its symbol: B
First time symbol B is seen, therefore a new cc_B is initialized
How all canonical circuits see this instance at start:
cc_A: Match/Recognition: 0.99999
cc_B: Match/Recognition: 0.99999
How all canonical circuits see this instance after learning:

[1]Log-sigmoid transfer function was also tried, without noticeable impact.

cc_A: Match/Recognition: 0.99995
cc_B: Match/Recognition: 0.99999

...

Instance 5:

Its symbol: C

First time symbol C is seen, therefore a new cc_C is initialized

How all canonical circuits see this instance at start:

cc_A: Match/Recognition: 0.99999
cc_B: Match/Recognition: 0.97186
cc_C: Match/Recognition: 0.99999

How all canonical circuits see this instance after learning:

cc_A: Match/Recognition: 0.83387
cc_B: Match/Recognition: 0.48339
cc_C: Match/Recognition: 0.87383

...

Instance 11:

Its symbol: A

cc_A already exists for it

How all canonical circuits see this instance at start:

cc_A: Match/Recognition: 0.93646
cc_B: Match/Recognition: 0.00132
cc_C: Match/Recognition: 0.04108

How all canonical circuits see this instance after learning:

cc_A: Match/Recognition: 0.98506
cc_B: Match/Recognition: 0.00015
cc_C: Match/Recognition: 0.04137

...

3.2. Cognitive Context Orthogonalizing Networks (CCON)

We recall that the goal of CCON is incremental and Bayesian orthogonalization of a cognitive context into symbols, each of which is maintained by a canonical circuit, executing in parallel with all other circuits. In computational terms only, CCON is a Bayesian classifier that can learn from an infinite data stream, with few instances at a time, and scale dynamically to any number of classes as they show up in the stream, processing each class in parallel.

CCON starts from a blank state, and adds a new canonical circuit for each new symbol when it first shows up in the data stream. The neural network

of a new canonical circuit is initialized using gradient descent so that $\xi_k = 1$ for the first data instance. Once a canonical circuit cc_k is initialized, its B_k maintains a conditional probability distribution for $\boldsymbol{\theta}^{(k)} = \{\theta_1^{(k)}, \theta_2^{(k)}, ...\theta_D^{(k)}\}$ where $\boldsymbol{\theta}^{(k)}$ represents the configuration of NN_k. B_k learns the distribution from the streaming data one instance at a time, while distancing itself from other symbols in the shared cognitive context.

To be more specific, let us suppose that new input $(\boldsymbol{x}, y)$ arrives at time t and we look at a single canonical circuit cc_k. The most recent probability distribution of $\boldsymbol{\theta}^{(k)}$ maintained by B_k is $p^{(t-1)}(\boldsymbol{\theta}^{(k)})$, last updated at time $t-1$, and NN_k is parameterized with a sample from it. The level of recognition $\xi_k(\boldsymbol{x}, \boldsymbol{\theta}^{(k)})$ can be used to measure the likelihood of $\boldsymbol{\theta}^{(k)}$ given $\boldsymbol{x}$ and y, based on a Bernoulli discriminative data likelihood model $\xi_k^y(1-\xi_k)^{1-y}$. To simulate inhibition between different canonical circuits, we formulate the data likelihood using a specialized set of i.i.d. instances for each canonical circuit. At time t, each canonical circuit cc_k has its own update set of instances $\mathfrak{D}_k^{(t)} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), ...(\boldsymbol{x}^{(K)}, y^{(K)})\}$, where each instance is the last seen of each of the K classes, and only $y^{(k)} = 1$ while all others are 0. Therefore all the $\mathfrak{D}_k^{(t)}$ share the same $\boldsymbol{x}^{(k)}$ but have different $y^{(k)}$. The data likelihood for cc_k at time t is then:

$$p(\mathfrak{D}_k^{(t)}|\boldsymbol{\theta}^{(k)}) = \prod_{i=1}^{|\mathfrak{D}_k^{(t)}|} p(y^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta}^{(k)}) = \prod_{i=1}^{|\mathfrak{D}_k^{(t)}|} \xi_k(\boldsymbol{x}^{(i)}, \boldsymbol{\theta}^{(k)})^{y^{(i)}}(1-\xi_k(\boldsymbol{x}^{(i)}, \boldsymbol{\theta}^{(k)}))^{1-y^{(i)}} \tag{4}$$

All the instances which have $y = 0$, i.e. all those associated with symbols of other canonical circuits, are used as a representation of what cc_k needs to distance from. In other words, the inhibition is being simulated by the $(1-\xi)^{1-y}$ factors in equation (4). It would be more biologically plausible if the model did not have to store even the last seen instance per symbol, and could learn with the current instance only. We believe this is possible if all the NN_k are modeled as recurrent neural networks and the desired inhibition between them is implemented through their attractors and synchrony. We attempted this approach but have so far been unsuccessful and this remains as key future work.

The Bayesian inference process executed in each cc_k can then be expressed as:

$$p^{(t)}(\boldsymbol{\theta}^{(k)}|\mathfrak{D}_k^{(t)}) \propto p^{(t-1)}(\boldsymbol{\theta}^{(k)})p(\mathfrak{D}_k^{(t)}|\boldsymbol{\theta}^{(k)}) \tag{5}$$

where $p^{(t-1)}(\boldsymbol{\theta}^{(k)})$ is the previous posterior, used as empirical prior at time

t. In addition to using only the last seen instance per symbol, one could optionally use multiple last seen instances per each symbol, and this improves the inhibition simulation.

As data stream instances keep coming in, each instance replaces the older instance of its symbol in all the $\mathfrak{D}_k^{(t)}$ sets, and for each new instance all canonical circuits update again in parallel using equation (5). The process recurrently reuses the last posterior as new empirical prior, developing $p(\boldsymbol{\theta}^{(k)})$ with each new instance, and constantly sampling from it for the latest NN_k parameter configuration.

So far we described how CCON learns. The model can also be used for predictions at any time by getting the recognition ξ_k of each of its canonical circuits and then using the softmax model:

$$p(y_{new} = k|\boldsymbol{x}_{new}) = \frac{\xi_k(\boldsymbol{x}_{new}, \boldsymbol{\theta}^{(k)})}{\sum_{l=1}^{K} \xi_l(\boldsymbol{x}_{new}, \boldsymbol{\theta}^{(l)})} \tag{6}$$

3.3. Incremental Metropolis Hastings (IMH)

We believe that a biologically plausible inference method for a single canonical circuit cc_k in CCON operates as follows at a high-level: it contains a generative process which constantly issues parameters for the neural network NN_k; while being constantly reconfigured, NN_k persistently tries to recognize its associated symbol in the streaming sensory data; supervision information is used to determine how well the recognition performs, thus measuring the likelihood of the proposed parameters; the current input persistently activates the neural network, and the generative process is able to keep proposing changes to the parameters while observing how their likelihood changes under the current input. Changes are accepted or rejected based on their likelihood effect and the neural network progresses through a chain of successively better configuration. The generative process can be imagined as turning a dial on each parameter with speed dependant on the likelihood effect, turning it faster when the likelihood gets worse, to look for a new value domain, and turning it slower when the likelihood improves, to optimize in the current value domain.

The method above is reminiscent of the Metropolis-Hastings (MH) Markov chain Monte Carlo algorithm [16]. However MH cannot learn from data streams because it requires all the data at each update step. Therefore, we develop IMH which updates with a single instance instead of all the data, and transfers the knowledge from previous updates through the prior. We

also adopt the acceptance criterion from MH by which proposals which improve the likelihood are always accepted, while proposals which degrade the likelihood can be tolerated to some extent for exploration benefits.

In this paper we only propose how IMH works on the representational Marr's level. How IMH is implemented on the physical Marr's level we leave as an open question. It is interesting to note however that we were led to the idea for IMH from our ongoing research into the physical level, where we are looking into a biological process which engulfs and manipulates many synapses inside a canonical circuit. This process has a wave-based dynamics and can be interpreted probabilistically as a generative process which continually samples neural network configurations, while learning from their performance.

Conventional MH builds a Markov chain which converges to a desired distribution $p^{(t)}$. It starts from a random value, and at each iteration proposes a new value θ_{new} from a proposal distribution q, to be a possible successor to the last chain element θ_{old}. The new value is accepted with probability

$$min(1, \frac{p^{(t)}(\theta_{new})q(\theta_{old})}{p^{(t)}(\theta_{old})q(\theta_{new})}) \tag{7}$$

Intuitively, θ_{new} has greater chances of acceptance if it is more probable in the target distribution $p^{(t)}$, and MH cares more for candidates which are harder to propose by q. The proposal distribution q could be a normal distribution centered on the previous value, and with a certain standard deviation. If the proposal is rejected, θ_{old} is copied as the new chain element. The empowering aspect of MH is that the normalizing constant of $p^{(t)}$ cancels out in equation (7). Because MH starts from a random value, the Markov chain needs to spend a certain time getting close to the target distribution before its values can be used as a representative sample. This initial part of the chain is often referred to as *burn-in* and removed after manual analysis.

IMH is based on the idea that instead of running a single long Markov chain requiring all the data for each update, we run a short update chain for each new data instance using only $\mathfrak{D}_k^{(t)}$ for data likelihood. Starting with a random value, each update chain at time t uses the previous posterior $p^{(t-1)}$ as an empirical prior, and an acceptance probability based on equations (5)

and (7):

$$min(1, \frac{p^{(t-1)}(\theta_{new})p(\mathfrak{D}_k^{(t)}|\theta_{new})q(\theta_{old})}{p^{(t-1)}(\theta_{old})p(\mathfrak{D}_k^{(t)}|\theta_{old})q(\theta_{new})} \tag{8}$$

The product of each update chain is an updated posterior $p^{(t)}$ in the form of a new set of samples. Since the posteriors used in IMH are sets of samples, we cannot use them in their original form as the next prior, since for a prior we need a functional form which can evaluate probability for any proposal θ_{new}. To bridge this gap we use kernel density estimation (KDE). Given a sample $\{\theta_1, \theta_2, ...\theta_S\}$ of size S from a probability distribution p, KDE can estimate the probability $p(\theta_{new})$ for any θ_{new}, by averaging the contributions of a symmetric kernel Λ centered on each point of the sample:

$$p_{KDE}(\theta_{new}) \approx \frac{1}{S}\sum_{s=1}^{S}\Lambda(\theta_s, \theta_{new}) \tag{9}$$

For Λ we use Gaussian kernels with bandwidth $(\frac{4}{3S})^{0.2}\sigma$, where σ is the sample standard deviation. This is a normal-optimal bandwidth heuristic [24], which we found allows for multiple modes, while preventing unbounded growth by merging proximal modes as their number grows.

Each update chain starts from a random sample from the previous posterior, thus using previous knowledge as a new starting point. Combined with the use of previous posterior as a very informative prior, a transfer of knowledge is achieved between update chains, allowing for much shorter chains (in our case, several orders of magnitude). The transfer of knowledge also removes the need for burn-in analysis, because after the first instance each update chain no longer starts with a random value, but in some vicinity of its target distribution.

IMH has simple parameterization because it automatically sets most parameters based on how well NN_k recognizes its symbol instances, and how well it rejects those of other symbols. To measure this quality we use the data likelihood from equation (4), and call it the *fitness* of a canonical circuit. The fitness is used to set the proposal distributions for the update chains, as well as their length.

For the proposal distribution q we use a Gaussian centered on the current value and with a standard deviation equal to the log of the fitness. As a result, the worse a circuit does on predicting a new instance (i.e. away

from its symbol or towards foreign symbols), the standard deviation will be exponentially larger, pushing it to propose alternatives more forcefully.

For the length of each update chain, IMH takes a minimum and maximum as parameters, and sets the actual length automatically as $maximum * (1 - fitness)$ with a lower bound of $minimum$. This giving less-fit canonical circuits longer chains and vice versa. Because of this, IMH goes through familiar instances faster, and slows down to deal with novelty.

The complexity of IMH is similar to MH because instead of running thousands of iterations using the whole dataset for each iteration, as is usually the case with MH, IMH can run much shorter chains for each data instance in sequence. IMH does have the minor added cost for calculating KDE for each update chain.

We recall that each canonical circuit cc_k learns independently, i.e. it runs its own IMH which is independent from the IMH processes of other canonical circuit. In terms of the update order of neural network parameters $\boldsymbol{\theta}^{(k)} = \{\theta_1^{(k)}, \theta_2^{(k)}, ...\theta_D^{(k)}\}$, they are updated one at a time and in random order permuted each iteration. We also assume independence of the prior components, i.e. $p(\boldsymbol{\theta}^{(k)}) = \prod_{j=1}^{D} p(\theta_j^{(k)})$.

4. Related Work

4.1. Biomorphic Perspective

Computational models of the neocortex vary greatly in their objectives and biological inspiration. Many of them do not pursue a generic function but are specialized models, for example in modeling ocular dominance with a 2D grid over which simple Hebbian-type relationships are simulated [25]. Models which are more generic cover only a subset of our five principal assumptions. Even if we ignore these assumption, existing models usually present either no evaluation or one on simplified datasets. In addition, they model either pattern completion or time-series predictions. If we consider, under the symbolic abstraction assumption, that a flow of reasoning can be seen as parallel cascades and co-occurrences of cognitive symbols, then we can argue that pattern completion and time-series predictions are relevant for the relationships between the different cognitive contexts, while a separate mechanism is still needed to select the symbol in each context. Since our work focuses on symbol identification in a cognitive context, it can therefore be seen as complementing to many existing models. Such complements can lead to models which combine symbolic and sub-symbolic (or connectionist)

computation, which has been considered as essential for artificial intelligence [26, 27]. Below we review these aspects through some key related work, in chronological order.

Fransen and Lansner [28] offer a model of associatively-networked *columns* using very biologically plausible neurons. Each of their columns is built from a single densely-connected recurrent neural network, and then the columns are interconnected and trained with a Bayesian correlation-based (Hebbian) learning rule. If these columns are assumed to be canonical circuits, then they do not provide symbol abstraction, but only operate sub-symbolic on raw input and transform it into a co-occurrence pattern. It is not clear if they are parallelizable. The model is evaluated only on a simple artificial pattern completion problem. This work provides a great starting point for using more biologically plausible neurons in our canonical circuits.

Maass et al. [29] present a system of recurrent networks of integrate-and-fire neurons, called Liquid State Machines, where they identify a column with each recurrent network. It is a type of reservoir computing, where each recurrent network is a reservoir which takes the input, forms transient dynamics, and offers opportunity for stable output neurons. In essence, it performs sophisticated input transformation. The output neurons can then be employed in some task like time-series prediction or classification. The use of the outputs however requires an additional method such as a perceptron-like local learning rule or linear regression, and it is unclear how these additional mechanisms relate to cortical circuits, or how they would be implemented biologically. Reservoirs are likely relevant to signal transformation in the brain, however by themselves do not offer a way for symbol abstraction. Perhaps the characteristic dynamics inside a reservoir could be related somehow to symbols, however this model has no method for this, and the authors simply say they would be "optimized genetically and through development". In addition, this model is not Bayesian.

Simen et al. [30] propose a neocortical model for a hybrid sub-symbolic and symbolic computation. Their equivalent of a canonical circuit is given the function of signal propagation delay and low-pass filtering, which is more of an auxiliary function, compared to our approach where the canonical circuit is an explicit representatives of a cognitive symbols. Their model then proposes that circuits can be organized with lateral inhibition as future work, but does not give any computational details. In addition there is no evaluation.

Nouri and Nikmehr [31] present a hierarchical Bayesian approach to reservoir computing with the objective of implementing a top-down Memory Prediction Framework (MPF). MPF essentially states that the system constantly tries to predict the future input based on its past experience. The proposed model is framed as a model of the neocortex and thalamus because of its hierarchical regions, and is strictly a time-series solution. The time-series prediction happens directly on raw sensory input, and there is no consideration for cognitive symbols. However, if the same idea is applied over our cognitive symbol probabilities instead of raw sensory inputs, then the regions in this model can relate to cognitive contexts, and it can be used for time-series predictions over cascades of contexts and symbols.

George and Hawkins [15] also present a hierarchical and temporal Bayesian model inspired by the columnar hypothesis, which learns temporal coincidences. As discussed above this perspective is relevant to the network of cognitive contexts. They do not address the issue of symbol abstraction from raw input inside a single context. Instead they skip this step and at the bottom layer present pre-identified symbols (Gabor filters). The authors state that "model would require modifications to include sparse-distributed representations within an HTM node", and this is where our model can be complementing.

Rinkus [2] presents a model for a generic function of canonical circuit, which agrees with our perspective that such a role becomes clear only in the context of its proximal peer group. It proposes that the proximal peer group stores sparse representation of the input in terms of which circuits are co-active in the group at the same time. The principal difference with our model is that there is no symbolic abstraction. Also, this work has no computational implementation or evaluation, is not Bayesian, and it uses only binary input attributes and binary synapses.

The most current, and perhaps most relevant to our model, is the work of Bastos et al. [13], which employs a Bayesian brain theory based on predictive coding and free energy minimization [32], and relates it remarkably to canonical circuits. Predictive coding is based on a hierarchical model where a higher area predicts the input of the lower area using a generative process, while the lower area computes the error between the prediction and the actual input, and sends back only the error to the higher area. The goal of the system is to minimize the error feedback, or the surprise, which requires better feedforward predictions. The model is conceptual only and there is no evaluation on datasets. It is also not clear how symbols relate on the same

level in the hierarchy or within a context. This presents an opportunity to combine this model with ours.

4.2. Computational Perspective

Due to the data stream learning requirements in our objective, we do not consider learning models which need all the data to be available at start. We should note however, that Metropolis-Hastings has been applied to learn neural network parameters given a set of training static data [33], using the complete data set for each chain update in order to determine whether to accept each proposal.

One related learning model is the work of de Freitas et al. [34], which, even though in much lower dimensional problems and mostly for time series analysis, uses particle filters to perform sequential Bayesian inference on neural network parameters. Therefore, we compare using particle filters in place of IMH as the Bayesian inference method for CCON. There are also other sequential Bayesian inference methods, however they are not applicable to our model due to its continuous state-space and the highly non-linear probability space of neural network parameters.

Particle filters are based on importance sampling which tackles an intractable distribution of θ by sampling from an approximating tractable distribution and weighting the samples with an importance weight w equal to the ratio of these two distributions. Unlike IMH which keeps a single parameter value at each moment, particle filters keep many versions (*particles*) of a parameter and use all of them based on their weights. Under a Markov process assumption this approach can be formulated recursively so that it can be used on streaming data. Starting with an initial set of approximating particles for θ, each iteration updates their weights and performs various additional techniques to help with a known particle degeneration issue where few particles end up with weights close to 1 while the rest end up close to 0. Aside from the degeneration issue getting worse with increasing model complexity, particle filters also inherit the limitation of importance sampling in scaling with higher dimensionality.

5. Evaluation

The objective of the evaluation is to see how the CCON works as a supervised classifier on data streams. Part of the evaluation is also to compare IMH to particle filters used as inference methods in CCON. It is a limitation

in our work that we do not evaluate on datasets which are more related to biological behavior. We are partially prevented from doing this because we would need a network of cognitive contexts, which is part of the future work.

We evaluate CCON in classification tasks on data streams using 10 datasets. Two of the datasets, *SEA* [35] and *Circles* [36], are popular for data stream evaluation and we use versions with 4 concept drifts each, at 25% intervals. The rest of the datasets are from the UCI repository[2]: *Waveform, Iris, Wine, Vehicle, Balance, Liver, Vertebral, and Diabetes*. The dataset *Waveform* is also popular in data stream evaluation however it has no concept drifts. The other UCI datasets are not data streams originally, however we shuffle them in a single fixed order and assume the dataset is presented one instance at a time. All data is normalized in the range [-1,1] before being used, as is customary before feeding data into neural networks.

For all experiments using random functions, 10 trials with different random initializations were used and their averages reported. Parallel programming was used to have all canonical circuits execute simultaneously. Accuracy was evaluated using the *prequential metric* [37] which is calculated by making a prediction on every new instance, comparing to the true label, and keeping a shifting sum of a loss function (0 for correct prediction and 1 otherwise). Its advantage over holdout validation is that it can be used on any data stream without losing training data, while it does converge to holdout estimates [37]. The shifting sum is over the last 50 instances, which is an evaluation (not model) parameter and chosen empirically, noting that evaluation results are not sensitive to it.

We present results for 3 and 7 hidden layer neurons in each canonical circuit's NN. Theoretically this parameter can also be made self-adjusting. The number of MH iterations for each update is self-adjusting based on minimum of 5 and maximum of 50. Setting any of these parameters does not require a parameter space search problem. They are all monotonic measures of allocated computational power. To calculate the data likelihood we try with only the last seen instance per symbol, or the last 7 per symbol to show how this impacts performance.

Figure 3 shows IMH significantly outperforming particle filters (p-value of 0.0001% in pairwise t-tests), when used as the incremental Bayesian inference method in CCON. The particle filters configuration shown is the best of 6

[2]http://archive.ics.uci.edu/ml/

versions which tried two different resampling techniques [38], sample sizes of 50 and 500, as well as 1 and 50 sample evolution iterations per data instance. For the transitional prior in our particle filters we use the same proposal distribution from our IMH update step.

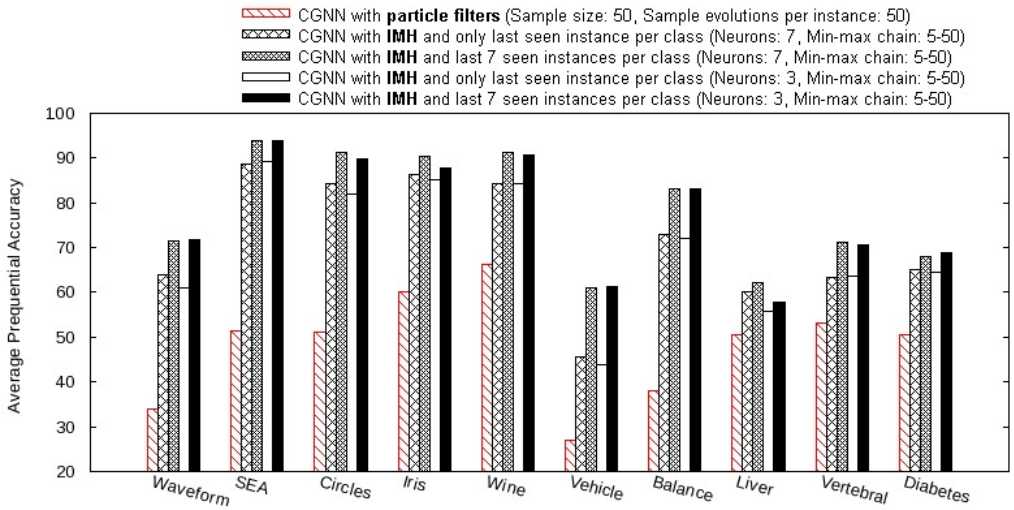

Figure 3: Comparing Bayesian inference methods for CCON. The left-most bar (in a diagonal pattern) shows the best particle filter configuration we found. The other bars (in crossed and solid patterns) show various configurations of IMH as detailed in the legend on top.

CCON is next compared, using IMH, to 4 leading methods for data stream learning: Hoeffding Adaptive Tree (also known as Very Fast Decision Tree), Naïve Bayes with Drift Detection Mechanism, and accuracy weighted ensemble classifiers using the above two methods as their base classifiers [37]. All methods have concept drift handling ability. Figure 4 shows that CCON is comparable with all of them, and that it excels on the *vehicle* dataset which is the most challenging in our evaluation. We see that CCON does well even with as little as 3 hidden neurons per symbol.

CCON is innately incremental because the full Bayesian inference at each update modifies the prior only where it does not fit with the new instance, preserving older and compliant knowledge. Figures 5 and 6 show that CCON handles concept drifts (marked with vertical lines) as good as any of the other methods.

While CCON matches the performance of state-of-the-art data stream

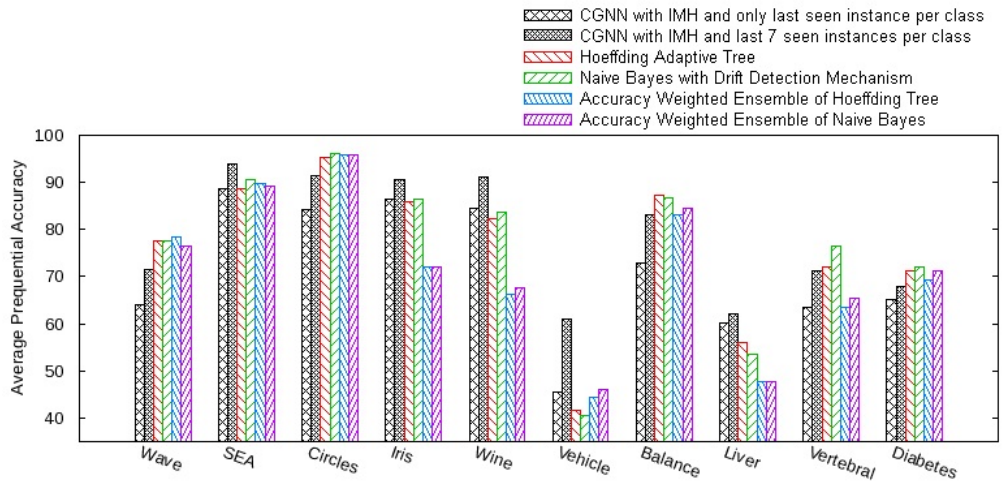

Figure 4: Comparing CCON to leading data stream learning methods. The left-most two bars (in crossed patterns) represent two IMH configurations: the left-most using only the last seen instance per symbol, and the other one using 7 previous per symbol. Both IMH configurations use a minimum of 5 and a maximum of 50 for the update chain length. The other bars (in diagonal patterns) show the other methods as described in the legend on top.

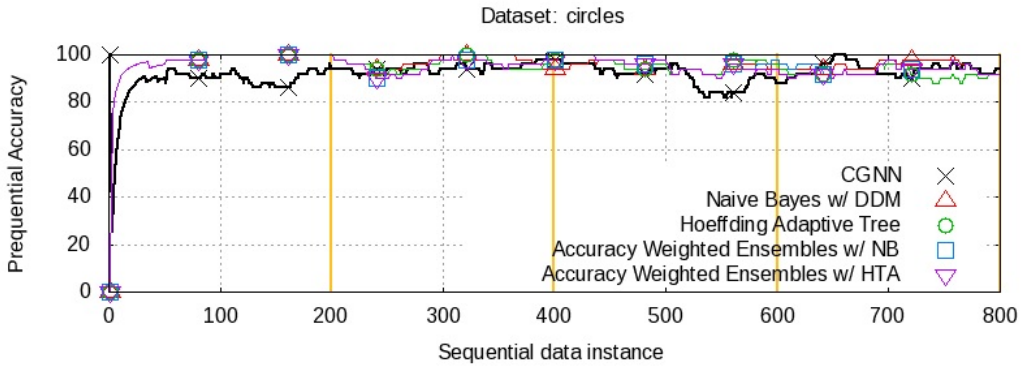

Figure 5: Concept drift handling on the *Circles* dataset. The symbol distributions change every 200 instances, at vertical lines.

classifiers, it has the following advantages: it is fully Bayesian and hence better with uncertainty, is not tied to a fixed number of classes and can add

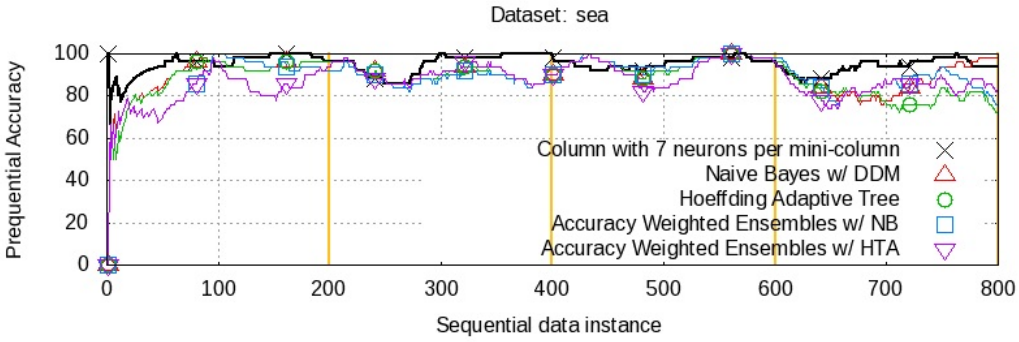

Figure 6: Concept drift handling on the *SEA* dataset. The symbol distributions change every 200 instances, at vertical lines.

classes as they show up in the data stream, has innate concept drift detection, and is much easier to parameterize. The other methods must keep a batch of old data or sufficient statistics and use special mechanisms to forget old knowledge in order to perform incremental learning. Therefore they require constants such as sliding windows, forgetting parameters, speed or bounds of change, all of which present their own parameter search problems [39].

6. Conclusion

We started by identifying five assumptions about neocortical computation: the existence of canonical circuits, their relationship to cognitive symbols and contexts, Bayesian aspects, parallelism and incremental requirements. Following these requirements we built our CCON model for a single cognitive context which is continually orthogonalized into symbols by canonical circuits, can follow the evolution of non-stationary symbols, and can grown the number of symbols dynamically as the context itself evolves.

We show that existing inference methods are insufficient to run the complexity of CCON. Following biological inspiration, we develop IMH as a novel inference method which enables CCON to perform as well as state-of-the-art data stream learners, evaluated on popular machine learning datasets. IMH is a contribution in itself as it can be used in other machine learning problems.

CCON covers only a single cognitive context. We are currently working on networking multiple cognitive contexts where they will co-supervise each

other. This will allows us to model higher level neocortical circuits and evaluate against biological behavior datasets.

References

[1] R. L. de Nó, J. F. Fulton, Architectonics and structure of the cerebral cortex, Oxford University Press, 1938.

[2] G. J. Rinkus, A cortical sparse distributed coding model linking mini- and macrocolumn-scale functionality, Frontiers in Neuroanatomy 4 (2010).

[3] V. B. Mountcastle, The columnar organization of the neocortex., Brain 120 (1997) 701–722.

[4] D. P. Buxhoeveden, M. F. Casanova, The minicolumn hypothesis in neuroscience, Brain 125 (2002) 935–951.

[5] J. Szentagothai, The ferrier lecture, 1977: the neuron network of the cerebral cortex: a functional interpretation, Proceedings of the Royal Society of London.Series B, Biological Sciences (1978) 219–248.

[6] S. Haeusler, W. Maass, A statistical analysis of information-processing properties of lamina-specific cortical microcircuit models, Cerebral Cortex 17 (2007) 149–162.

[7] A. M. Bastos, W. M. Usrey, R. A. Adams, G. R. Mangun, P. Fries, K. J. Friston, Canonical microcircuits for predictive coding, Neuron 76 (2012) 695–711.

[8] H. Markram, M. Toledo-Rodriguez, Y. Wang, A. Gupta, G. Silberberg, C. Wu, Interneurons of the neocortical inhibitory system, Nature reviews. Neuroscience 5 (2004) 793–807.

[9] N. M. da Costa, K. Martin, Whose cortical column would that be?, Frontiers in Neuroanatomy 4 (2010).

[10] J. DeFelipe, H. Markram, K. S. Rockland, The neocortical column, Frontiers in Neuroanatomy 6 (2012).

[11] S. Harnad, The symbol grounding problem, Physica D: Nonlinear Phenomena 42 (1990) 335–346.

[12] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, N. D. Goodman, How to grow a mind: Statistics, structure, and abstraction, Science 331 (2011) 1279–1285.

[13] A. M. Bastos, W. M. Usrey, R. A. Adams, G. R. Mangun, P. Fries, K. J. Friston, Canonical microcircuits for predictive coding, Neuron 76 (2012) 695–711.

[14] K. J. Friston, K. E. Stephan, Free-energy and the brain, Synthese 159 (2007) 417–458.

[15] D. George, J. Hawkins, Towards a mathematical theory of cortical micro-circuits, PLoS Computational Biology 5 (2009).

[16] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin, Bayesian data analysis, Chapman and Hall, 2004.

[17] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[18] D. Marr, Vision: a computational investigation into the human representation and processing of visual information, 1982.

[19] E. Vul, N. D. Goodman, T. L. Griffiths, J. B. Tenenbaum, One and done? optimal decisions from very few samples, in: Proceedings of the 31st annual conference of the Cognitive Science Society, volume 1, 2009, pp. 66–72.

[20] A. N. Sanborn, T. L. Griffiths, D. J. Navarro, A more rational model of categorization, in: Proceedings of the 28th annual conference of the Cognitive Science Society, 2006, pp. 726–731.

[21] R. Rao, Hierarchical bayesian inference in networks of spiking neurons, in: Advances in neural information processing systems, 2004, pp. 1113–1120.

[22] S. Deneve, Bayesian inference in spiking neurons, in: Advances in neural information processing systems, volume 17, 2005, pp. 353–360.

[23] B. C. Csáji, Approximation with artificial neural networks, Master's thesis, Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[24] A. W. Bowman, A. Azzalini, Applied Smoothing Techniques for Data Analysis : The Kernel Approach with S-Plus Illustrations: The Kernel Approach with S-Plus Illustrations, OUP Oxford, 1997.

[25] G. Goodhill, Topography and ocular dominance: a model exploring positive correlations, Biological Cybernetics 69 (1993) 109–118.

[26] M. L. Minsky, Logical versus analogical or symbolic versus connectionist or neat versus scruffy, AI Magazine 12 (1991) 34.

[27] J. Wallace, K. Bluff, Neurons, glia and the borderline between subsymbolic and symbolic processing, Progress in Artificial Intelligence 990 (1995) 201–211.

[28] E. Fransen, A. Lansner, A model of cortical associative memory based on a horizontal network of connected columns, Network 9 (1998) 235–264.

[29] W. Maass, T. Natschlager, H. Markram, Real-time computing without stable states: A new framework for neural computation based on perturbations, Neural Computation 14 (2002) 2531–2560.

[30] P. Simen, T. Polk, R. Lewis, E. Freedman, Universal computation by networks of model cortical columns, in: Proceedings of the International Joint Conference on Neural Networks, volume 1, IEEE, 2003, pp. 230–235.

[31] A. Nouri, H. Nikmehr, Hierarchical bayesian reservoir memory, in: Proceedings of the 14th International Computer Conference (CSICC), IEEE, 2009, pp. 582–587.

[32] K. J. Friston, K. E. Stephan, Free-energy and the brain, Synthese 159 (2007) 417–458.

[33] R. M. Neal, Bayesian Learning for Neural Networks, lecture notes in statistics no. 118 ed., Springer-Verlag, New York, 1996.

[34] J. F. G. de Freitas, M. Niranjan, A. H. Gee, Hierarchical bayesian models for regularization in sequential learning, Neural Computation 12 (2000) 933–953.

[35] W. N. Street, Y. Kim, A streaming ensemble algorithm (sea) for large-scale classification, in: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2001, pp. 377–382.

[36] K. Nishida, K. Yamauchi, Detecting concept drift using statistical testing, Discovery Science (2007) 264–269.

[37] J. Gama, Knowledge Discovery from Data Streams, 1st ed., Chapman and Hall, 2010.

[38] R. V. D. Merwe, A. Doucet, N. D. Freitas, E. Wan, The unscented particle filter, in: Advances in neural information processing systems, 2001, pp. 584–590.

[39] A. Bifet, R. Gavalda, Adaptive learning from evolving data streams, Advances in Intelligent Data Analysis VIII (2009) 249–260.