Cognitive Architectures for Sensory Processing

Jose C. Principe, *Fellow, IEEE* and Rakesh Chalasani, *Student Member, IEEE*

**Abstract**

This paper describes our efforts to design a cognitive architecture for object recognition in video. Unlike most efforts in computer vision, our work proposes a Bayesian approach to object recognition in video, using a hierarchical, distributed architecture of dynamic processing elements that learns in a self-organizing way to cluster objects in the video input. A biologically inspired innovation is to implement a top-down pathway across layers in the form of causes, creating effectively a bidirectional processing architecture with feedback. To simplify discrimination, overcomplete representations are utilized. Both inference and parameter learning are performed using empirical priors, while imposing appropriate sparseness constraints. Preliminary results show that the cognitive architecture has features that resemble the functional organization of the early visual cortex. One example showing the use of top-down connections is given to disambiguate a synthetic video from correlated noise.

**Index Terms**

Visual Cortex, Empirical Bayes, Object Recognition, Top-Down.

**I. INTRODUCTION**

The ability of organisms to interact with the world is unparalleled and has been the subject of intensive studies in both neuroscience and engineering. For roboticists interested in building autonomous systems, effective processing of sensory information is critical exactly for the same reasons as those of biological organisms: to navigate successfully through complex time varying environments. However, sensory cortices are still superior to the most advanced systems built...
by engineers. Engineering systems can be made very accurate for very specific tasks, but they
tend to fail at more general tasks and in unknown environments.

Most work in image recognition (and audio too) still utilizes sensory inputs to design features
that describe the visual environment [1]. This is the obvious choice because designers can
use their intuition and experience to find good features in the signals they collect. However,
such hand-engineered features tend to be very task specific and do not generalize well. Many
years have been spent to find proper features to translate the complexity and invariances of the
real world, but the problem is still largely unsolved. We departed from this reasoning several
years ago by hypothesizing that what matters to an autonomous agent is the invariance of
its internal representations—not the feature invariance [2]. So we believe the effort should
be directed towards creating computational systems that explain the world in its many forms
using rich internal spaces where representations can be made stable and discriminative for fast
and precise recall. This was basically Helmholtz’s reasoning when he stated “the perceptual
system is an inference engine whose function is to infer the probable causes of the sensory
input” [3]. Bayesian inference appears powerful enough to solve this computational problem by
appealing to a generative model and using approximate Bayesian inference. However, one has to
carefully parameterize the posterior probability distributions that incorporate a priori knowledge
and uncertainty about the causes of sensations. This parameterization becomes important for
efficient inversion of the generative model—particularly in approximate Bayesian inference of
the sort that we will consider.

The first thing that one realizes when studying cognition is that perception is not passive as
in computer vision or audition. Perception is an active process, where literally one “sees” what
one wants to see, i.e. the goal is to search the sensory input to corroborate or falsify the internal
predictions, explaining in a global way the input data, and creating the wholeness of conscious
experience. Fuster [4] perhaps best presents the epitome of this approach when he declares,
“Perception is memory updating”. This creates an enormous chicken and egg problem that we
do not yet know how to solve because it means that to perceive requires past experience, and it
is quite unclear how memories are created in the first place. A cursory analysis of the famous
visual system diagram by Van Essen [5] (Fig. 1a) shows the distributed, hierarchical, feedback
rich architecture that evolved to process the spatio-temporal information existing in the external
world. It is also worth pointing out that the hippocampus is on top of the hierarchy, suggesting
to us engineers that the role of the visual system is to extract the unexpected information from the current scene that will be then consolidated in new memories that the organism can later use in a more efficient way for quick action. If the goal is to emulate the inner workings of the visual cortex, we submit that our computational models should be distributed, hierarchical, and online including both bottom-up and top-down processing, and with parameters that are learned from data in a self-organized manner (Fig. 1b).

This architecture is in stark contrast with our conceptual models of computation that have been inspired by the powerful universality of the Von Neumann architecture [6], where memory is relegated to a static, subsidiary role of storage for code and data. Unfortunately, this formal modeling is still the widely accepted view in cognitive science, and our work departs from these assumptions on the role of the memory too [7]. We seek to develop a computational architecture where goals and/or memory (past experience) from the top levels can be used locally as “time signals” and mixed with incoming sensory input at all hierarchical processing levels. We venture to call this architecture cognitive because it learns autonomously to represent the external environment from sensory data, stores, and uses this knowledge to infer invariant representations at different spatial and temporal scales that later are used to disambiguate the objects in the environment.

In order to mathematically program such an approach, dynamic systems were selected as the building block of our computational architecture. Dynamic systems allow for two types of memory: long-term in its parameters and short-term in its states. The advantage of a dynamic system framework is that it naturally handles time, uses directly functional mappings, encodes uncertainty, and can learn multiple internal variables in an online framework. Dynamics are also important to exploit and impose constraints that translate the smooth changes in natural visual or acoustic scenes. With dynamic models we believe we can fulfill the design constraints of on-line in situ computation i.e. where memory, goals, and computation are mixed efficiently.

In addition, we create a distributed, hierarchical computational model where the expected goals are coded in top-down signals, which we will call *causes*, and are combined in the dynamic models along with the bottom-up signals from the sensors. Finally, causes, which reflect past experience, have to be also handled and stored by the autonomous system in a principled way. Again, using ideas from computational neuroscience, content addressable memories can store the top layer causes as gestalts that translate past experience into signals that can be used in the
top-down processing as priors to disambiguate the incoming sensory input.

In this paper, we will associate online inference about the latent or hidden causes of sensory input with perceptual inference. Perceptual inference is distinct from learning, which we associate here with the optimization of the parameters of a generative model that determines how causes generate sensory signals (and each other). Later we will use a two-step optimization scheme, in which latent causes and states are updated to optimize an energy function to simulate inference, while slowly varying parameters learn structural regularities in the generation of those signals by optimizing exactly the same objective function.

II. A COMPUTATIONAL ARCHITECTURE FOR OBJECT RECOGNITION IN VIDEO

The visual system as described above can be implemented as a Bayesian generative model that learns the external environment using a set of parameters in an unsupervised way. Using these parameters, the model infers the underlying cause that might have generated the observations. Here we discuss one such generative model based on predictive coding or empirical Bayes [8],[9].
The basic idea of predictive coding is to model the visual system as a hierarchical generative model that attempts to predict the external responses using several layers of processing units and learn the system that suppresses the prediction error at all the levels. If the generative model is able to reproduce the input stimuli accurately then it means the model has learned to explain these inputs. Recognition is then simply solving an inverse problem by jointly minimizing the prediction error at all levels. This framework forms the theoretical underpinning of the methods proposed in this work and we argue here that such model encompasses the vast majority of the existing methods, including several deep learning networks. The basic building block of our proposed model that is pervasive across all the layers is a generalized state-space model with additive noise, as in predictive coding [10]:

\[
\begin{align*}
    y_t &= F(x_t, u_t) + n_t \\
    x_t &= G(x_{t-1}, u_t) + v_t
\end{align*}
\]

where \( y_t \) is the data, \( F \) and \( G \) are functions parameterized by \( \theta \), and the terms \( u_t \) are called the unknown causes. Since we are interested in obtaining abstract information from observations, the causes are encouraged to have a non-linear relationship with the observations. The hidden states \( x_t \) mediate the influence of the cause on the output and endow the system with short-term memory [10]. Several such blocks can now be stacked, such that the output from one layer acts as an input to the layer above, to form a hierarchical model. This takes the form of an L-layered network:

\[
\begin{align*}
    u^{(l-1)}_t &= F(x^{(l)}_t, u^{(l)}_t) + n^{(l)}_t \\
    x^{(l)}_t &= G(x^{(l)}_{t-1}, u^{(l)}_t) + v^{(l)}_t
\end{align*}
\]

As the causes are fed into the next layer as input, the causes link the layers while the states link the dynamics over time. The important point in this design is how the higher-level predictions influence the inference at lower levels. The predictions at the higher layer non-linearly enter into the state space model at a bottom layer by empirically altering the prior on the causes of the bottom layer. Hence, these top-down connections, along with the horizontal connections in the state space, directly influence the inference in the bottom layers. The proposed architecture has several key features that exploit and extend predictive coding or empirical Bayesian schemes, such as:
- Bidirectional (top-down and bottom-up) in situ processing that enables the model to be controlled both by sensory data and (empirical) beliefs about causes from higher layers to optimize perceptual inference.

- Dynamics in the visual inputs are considered. This allows beliefs about temporal context to influence perceptual inference.

- Only salient features of the input data are stored by the model; hence the causes form a compressed and sparse representation.

- The reutilization of the same model within and across layers brings the possibility of reusable modules to be efficiently implemented in hardware.

In the predictive coding framework, the basic building block is a state-space model of the form given by (1). One can show that this state-space model encompasses several popular existing methods for object recognition [10][11]. In other words, from the Bayesian network perspective this can act like a universal generative model. The ability of these models to generalize several models comes by assuming different kinds of prior knowledge on the states and the causes and, also, the design while encoding the inputs. For example, sparse coding [12], which is popularly used in object recognition, can be considered as a special case of the state space models when we do not have the hidden states and a sparsity inducing prior is employed. This sparsity can be implemented explicitly in terms of assumptions about random fluctuations in the generative model (such as using $\ell_1$-norms—see below). Alternatively, one can implement sparse distributions using a non-linear function of the hidden causes implicit in $g(x_t, u_t)$ [13]. Even more complex methods like independent subspace analysis (ISA) (or topographic ICA) [14][15], where hidden states for feature extraction and causes that pool from these states with fixed weights, are a special case albeit without any dynamic state-space equation. Recently, restricted Boltzmann machines [16], auto-encoders [17], encoder-decoder models [18] have become popular to build deep networks. The key to these models is to learn both encoding and decoding concurrently and to build the deep network as a feed forward model using only the encoder while discarding the decoder. Though these models appear to be different from the hierarchical models described below, the function of the encoder is only to approximate the decoding (or inference) and hence, can be subsumed in the same framework as well. Similarly, many other popular deep networks used for object recognition contains max or average pooling functions along with feature extraction
methods [19]. These can also be subsumed into the proposed model by considering the states as encoded features, while the non-linear function between the states and the causes is simply a max or averaging pooling function.

Previously, many methods—like slow feature analysis [20], temporal ISA [21]—used temporal coherence to learn parameters of the model, that are eventually useful for object recognition. Also, Memisevic and Hinton [22] (conditional RBM) and Cadieu and Olshausen [23] proposed methods to learn short-term temporal relationships. However, none of these models consider top-down connections and hence, do not incorporate any expected goal (or use past experiences) during perceptual inference.

III. Computational Model

From the discussion above, we now have a general framework to build a deep learning network. However, to parse out the complexity of the external world, the model needs to extract discriminative information. In the following sections we describe the procedure to build one such deep network that is appropriate for visual object recognition. In line with the general state space model presented above, each layer consists of two modules: a state space model and an invariance-learning model. The overall hierarchy is built in a self organizing way using a layer-wise learning procedure accessing only the bottom-up information coming from the layers below, similar to other deep learning methods [24]. After learning the parameters at one layer, the parameters are fixed and the output of that layer is fed into higher layer as inputs. In the following sections we explain the architecture and the learning procedure in more detail.

A. Single Layer Module

Let \( \{y_1, y_2, \ldots, y_t\} \in \mathbb{R}^p \) be a \( p \)-dimensional sequence of a 2-D patch extracted from the same location across all video frames. To model this patch, a network consisting of two distinctive parts is built (see Fig. 2a): feature extraction (inferring states) and pooling (inferring causes). For inferring states \( x_t \in \mathbb{R}^k \), a linear state space model with sparsity constraint—referred to as dynamic sparse coding (DSC)—on the states will be utilized. More formally, we map the inputs \( y_t \) at time \( t \) onto an over-complete dictionary of \( k \)-filters, \( C \in \mathbb{R}^{p \times k} (k > p) \), while keeping track of the dynamics in the latent states with state-transition matrix \( A \in \mathbb{R}^{k \times k} \). The negative
Fig. 2: (a) Shows a single layered network on a group of small overlapping patches of the input video. The green bubbles indicate a group of inputs $(y^{(n)}_t, \forall n)$, red bubbles indicate their corresponding states $(x^{(n)}_t)$ and the blue bubbles indicate the causes $(u_t)$ that pool all the states within the group. (b) Shows a two-layered hierarchical model constructed by stacking several such basic blocks. For visualization no overlapping is shown between the image patches here, but overlapping patches are considered during actual implementation.

log-likelihood or energy function for DSC can be written as:

$$-\log P(y_t, x_t|C, A) = E_1(x_t, y_t, C, A)$$
$$= \|y_t - Cx_t\|^2_2 + \lambda \|x_t - Ax_{t-1}\|_1 + \gamma \|x_t\|_1$$  \hspace{1cm} (3)

Notice that the second term involving the state-transitions is also constrained to be sparse to make the state-space representation consistent.

The second part of the model involves learning invariant representations by taking advantage of the spatial relationships in a local neighborhood. Towards this goal, a small group of states $x^{(n)}_t$, where $n \in \{1, 2 \ldots N\}$ represents a set of contiguous patches w.r.t position in the image space, are added (or sum pooled) together to facilitate local translation invariance. Furthermore, $d$-dimensional causes $u_t \in \mathbb{R}^d$ are inferred from the pooled states to obtain a representation that is invariant to more complex local transformations like rotation, spatial frequency, etc [25]. Specifically, the causes $u_t$ are inferred by minimizing the following energy function with an $\ell_1$
regularization that promotes sparsity:

\[- \log P(x_t, u_t | B) = E_2(u_t, x_t, B) = \sum_{n=1}^{N} \left( \sum_{k=1}^{K} |\gamma_k \cdot x_{t,k}^{(n)}| \right) + \beta \|u_t\|_1 \]  

(4)

\[ \gamma_k = \gamma_0 \left[ 1 + \exp\left(-[Bu_t]_k\right) \right] \]

where \( \gamma_0 > 0 \) is some constant. Notice that here \( u_t \) multiplicatively interacts with the accumulated states through \( B \), modeling the shape of the sparse prior on the states. Essentially, the invariance matrix \( B \) is adapted such that each component \( u_t \) connects to a group of elements in the accumulated states that co-occur frequently. In other words, whenever a component in \( u_t \) is active it lowers the coefficient of a set of components in \( x_{t}^{(n)} \) \( \forall n \), making them more likely to be active. Since co-occurring components typically share some common statistical regularity, such activity of \( u_t \) leads to a locally invariant representation [25].

These components of the energy functionals form a generative model specified as the log probability of obtaining observations, their hidden causes and the model parameters.

\[- \log P(y_t, x_t, u_t, \theta) = - \log P(y_t | x_t, u_t, \theta) - \log P(x_t, u_t | \theta) - \log P(\theta) \]

\[ \Rightarrow E(y_t, x_t, u_t, \theta) = \sum_{n=1}^{N} \left( \frac{1}{2}\|y_t^{(n)} - Cx_t^{(n)}\|_2^2 + \lambda\|x_t^{(n)}\|_1 + \sum_{k=1}^{K} |\gamma_{t,k} \cdot x_{t,k}^{(n)}| \right) + \beta \|u_t\|_1 \]

\[- \log P(\theta) \]  

(5)

where \( \gamma_{t,k} = \gamma_0 \left[ 1 + \exp\left(-[Bu_t]_k\right) \right] \) and \( \theta = \{A, B, C\} \)

Inference and learning corresponds to minimizing this energy functional with respect to the latent variables \((x_t, u_t)\) and parameters \((\theta)\), respectively. In approximate Bayesian inference schemes, this energy functional takes the form of variational free energy [10]. The latent variables \((x_t, u_t)\) can be inferred efficiently using first-order based proximal gradient methods that minimize this energy function. The inference procedure is discussed more thoroughly in our previous work [26].

1) Learning in the Single Layered Network: The model parameters \((\theta = \{A, B, C\})\) are learned by performing dual estimation filtering [27] (similar to block coordinate descent) on the combined cost function (5), where inferring the latent variables \((x_t, u_t)\) alternates with parameter
updating, while the other is held fixed. Since the inputs here are a time-varying sequence, the parameters ($\theta$) are updated using gradient descent with an additional temporal smoothness prior:

$$\theta_t = \theta_{t-1} + z_t \quad (6)$$

where $z_t$ is Gaussian transition noise over the parameters. This keeps track of their temporal relationships and acts as a momentum term during the gradient update. Notice that with fixed $x_t$ and $u_t$, each of the parameter matrices can be updated independently, whose gradient is obtained as follows:

$$\nabla_A E = \text{sign}(x_t - A_t x_{t-1}) x_t^T + \zeta(A_t - A_{t-1})$$

$$\nabla_C E = (y_t - C_t x_t) x_t^T + \zeta(C_t - C_{t-1}) \quad (7)$$

$$\nabla_B E = (\exp\{-[B u_t] \cdot |x_t|\} u_t^T + \zeta(B_t - B_{t-1})$$

where $\zeta$ is called the forgetting factor. Matrices $C$ and $B$ are column normalized after the update to avoid any trivial solution.

IV. Multi-layered Architecture

The architecture of the multilayered processing model is a tree structure, with the simple encoding module described in Section III-A replicated at each node of the tree (Fig. 2b). At the bottom layer, the nodes are arranged as a tiling of the entire visual scene and the parameters across all the nodes are tied, resembling a convolution over the input frame. Each node encodes a small patch of the input video sequence, which is useful for parallelizing the computation. The computational model is uniform within a layer, and across layers, albeit with different dimensions; the only thing that changes is the nature of the input data. Note that within each block the features extracted from a spatial neighborhood are pooled together, indicating a progressively increasing receptive field size of the nodes with the depth of the network. For this reason, we also expect that the feature activations in the higher layers change slowly over time than the lower layer activations [28]. Parameter learning at each layer uses a greedy layer-wise procedure, i.e. the parameters at the bottom layer modules are learned first from a sequence of small patches extracted from the input video sequences; only after these parameters are learned does learning of the next layer begins. Fig. 3 exemplifies the inference on a two-layer network with a single module in each layer for simplicity. Here the layers in the hierarchy are arranged in a Markov
chain, i.e., the variables at any layer are only influenced by the variables in the layer below and the layer above. Specifically, at the bottom layer for example, sequences of patches ($y_t$) extracted from fixed spatial locations spread across the entire 2D space of the video frames is fed as input to each first layer modules. On the other hand, the top-down predictions of the first layer causes coming from the second layer try to modulate the representations. The bidirectional nature of the model is apparent in this figure, and in general there may be an extra top-down predictions as input to provide context for the analysis. Next, we will include the modifications in the general form of the generative model to exploit high order structure in the generation of observations, in terms of empirical priors.

**V. Inference in Multi-Layered Network with Top-Down Connections**

With the parameters fixed, inferring latent variables at any intermediate layer involves obtaining useful representations of the data driven bottom-up information while combining the top-down influences from the higher layers. In other words, while the hierarchical representations try to extract useful information from the inputs for recognition, the top-down connections modulate the representations at each level with abstract knowledge from the higher layers. As we will show next, the top-down connections “convey” contextual information to endow the model with a prior knowledge for extracting task specific information from noisy inputs.

In the hierarchical model, the energy function at any layer ($l$) that needs to be minimized to
infer $x^{(l)}_t$ and $u^{(l)}_t$ is given by:

$$E_t(x^{(l)}_t, u^{(l)}_t, \theta^{(l)}) = \sum_{n=1}^{N} \left( \frac{1}{2} \|u^{(l-1,n)}_t - C^{(l)} x^{(l,n)}_t\|_2^2 + \lambda \|x^{(l,n)}_t - A^{(l)} x^{(l-1)}_t\|_1 ight) + \sum_{k=1}^{K} [\gamma^{(l)}_{t,k} \cdot x^{(l,n)}_t] + \beta \|u^{(l)}_t\|_1 + \frac{1}{2} \|u^{(l)}_t - \hat{u}^{(l+1)}_t\|_2^2$$

(8)

$$\gamma^{(l)}_{t,k} = \gamma_0 \left[ \frac{1 + \exp\left(-[B^{(l)} u^{(l)}_t]_k \right)}{2} \right]$$

where $u^{(l-1)}_t$ are the causes from the layer below and $\hat{u}^{(l+1)}_t = C^{(l+1)} x^{(l+1)}_t$ are the top-down prediction, reflecting the states of the layer above. Note that the last term in the energy function $\left( \frac{1}{2} \|u^{(l)}_t - \hat{u}^{(l+1)}_t\|_2^2 \right)$ is in fact the observation equation for the layer above and influences the representation at the $(l)$th layer by reducing the top-down prediction error. In other words, the goal is to match the representation of the inputs from the layer below with the belief of the layer above about the same representation. We have limited experience with the role of this top-down causes, but we found that it can stabilize the learned representations in noisy environments. More generally, the top-down causes can come from a memory block storing prior experience and provide guidance to disambiguate inputs or select one object among others. Such top-down modulation that selectively parses out irrelevant features is called feature based attention [29][30].

VI. Experiments

A. Learning from Natural Image Sequences

In the following experiments we show that the working of the states and causes at each layer resemble that of simple and complex cells in the visual cortex, respectively. The states act as simple feature detectors, while causes encode complex invariances. However, the key to our model is that the responses of both the states and the causes are influenced by the context, coming from both temporal and top-down connections, making them capable of representing inputs that are beyond their characteristic receptive fields.

Firstly, we consider learning a two-layer model from natural video sequences obtained from Van Hateren’s video database [31]. This database contains several video clips of natural scenes containing animals, trees, etc. and each frame of these video clips is preprocessed using local contrast normalization as described in [19]. Sequences of patches are then extracted from the preprocessed video sequences to learn the parameters of the model.
Fig. 4: Receptive fields of the model at different levels. (a) Shows the receptive fields of the states in Layer 1, i.e., the columns of the observation matrix $C^{(1)}$. (b) and (c) shows the receptive fields of the layer 1 and layer 2 causes, respectively. The receptive here are constructed as a weighted combination of the columns of the layer 1 observation matrix $C^{(1)}$.

For the first layer, we use $17 \times 17$ pixel patches to learn 400 dimensional states and 100 dimensional causes. We take the pooling between the states and the causes to be $2 \times 2$, i.e., we further divide each of the $17 \times 17$ patches into 4 overlapping $15 \times 15$ pixel patches and the states extracted from each of these subdivided patches are pooled to obtain the causes (refer to Fig. 2). Now, for the second layer, we take $20 \times 20$ pixel patches to learn 200 dimensional states and 50 dimensional causes. Here we again use $2 \times 2$ pooling between the states and the causes, i.e., each of the $20 \times 20$ pixel patches is divided into 4 overlapping $17 \times 17$ patches from which first layer causes are inferred. These first layer causes act as inputs to the second layer. States are obtained from each of these 4 inputs and pooled together to obtain the causes of the second layer.

Fig. 4 shows the receptive fields of the states/causes at different levels of the learned model. At the bottom layer, the receptive fields of the states are simple localized lines resembling the Gabor functions, while that of the causes resemble grouping of these primitive feature, localized in orientation and/or spatial position (more about this is Section VI-A1). Whereas, the receptive fields of the second layer causes resemble even more complex features like corners, angles, etc. These filters are consistent with the previously proposed methods like Lee et al. [32] and Zeiler et al. [33], although they have been extracted from video scenes.
Fig. 5: Visualizing invariance represented by first layer causes. Firstly, each dictionary element (with a receptive field of $15 \times 15$ pixels) in the first layer observation matrix ($C^{(1)}$) is fit with a Gabor function and is parametrized in terms of the center position, spatial frequency and orientation of the Gabor functions. (a) (Top row) Shows the scatter plot of the center positions of the dictionary elements along with the orientations and (Bottom row) shows the polar plot of orientations versus spatial frequency. (b) Shows the connection strength between the first layer invariance matrix ($B^{(1)}$) and the observation matrix ($C^{(1)}$), i.e., it shows which subset of the dictionary elements are most likely active when a column of the invariance matrix is active. Each box represents one column of the invariance matrix $B^{(1)}$ and 10 out of 100 columns are randomly selected. Part (b) shows the center and orientation scatter plots (top) and the corresponding spatial frequency and orientation polar plots (bottom), highlighting the most active dictionary subset elements for a selected $B^{(1)}$ column. Notice, per each active column in $B^{(1)}$, a subset of the dictionary elements (not unique) is grouped together according to orientation and/or spatial position, indicating invariance to the other properties like spatial frequency and center position.

1) Visualizing Model Invariance: To get a better understanding of the invariance learned by the model, we visualize the connections between the first layer states and the causes. Fig. 5 shows the results obtained—see caption for more details. We observe that most of the columns
Fig. 6: Shows the temporal structure learned by the model. (a) Depicts the connection strength (of matrix $A^{(1)}$) between the layer 1 state elements over time. Read the figure as follows: if the basis on the left is active at time $t$ (presynaptic), then the corresponding set of bases on the right indicates the prediction for time $t + 1$ (postsynaptic). This indicates that certain properties, like orientation and spatial positions, change smoothly over time. (b) Shows the scatter plot of the 15 strongest connections per each element in the matrix $A^{(1)}$, arranged according to the orientation selectivity of the pre and postsynaptic elements. Notice that most points are within $\pi/6$ from the diagonal, indicated by the black lines.

of the invariance matrix group together dictionary elements that have similar orientation and frequency, while being invariant to the other properties like translation. However, there are other types of the invariances as well, where the dictionary elements are grouped only by spatial location while being invariant to other properties.

2) Learning Temporal Structure: As shown above, the receptive fields of the bottom layer states in our model resemble that of simple cells in the V1 area of the visual cortex. It is now well known that these cells act as simple oriented filters and strongly respond to particular inputs [12]. However, recent studies show that their influence extends beyond their receptive fields [34], modulating the response of other cells, in both excitatory and inhibitory ways, depending on the spatial and temporal contextual information. In our approach, such temporal context at each layer
is modeled using the parameter matrix $A^{(l)} \forall l$. Fig. 6 shows a visualization of this matrix at the bottom layer. We observe that the model learns to maintain certain properties, like orientation and spatial position, over time. In other words, given that a basis is active at a particular time, it has excitatory connections with a group of other bases (sometimes with strong self-recurrence connections), making them more likely to be active at the next time instance. On the other hand, along with the sparsity regularization, it also inhibits the response of other elements that are not strongly connected with the active basis over time.

**TABLE I: Classification results (%) obtained over COIL-100 and Animal datasets.**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SC</th>
<th>DSC</th>
<th>DSC-I</th>
<th>ConvNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>COIL-100</td>
<td>66.87</td>
<td>71.81</td>
<td><strong>74.63</strong></td>
<td>71.49 [35]</td>
</tr>
<tr>
<td>Animal</td>
<td>76.09</td>
<td>82.34</td>
<td><strong>85.82</strong></td>
<td>—</td>
</tr>
</tbody>
</table>

3) **Recognition with Temporal Context:** As discussed above, the temporal context helps to maintain properties like orientation and spatial location consistent in the representations over time, while causes maintain invariance to a similar set of properties. Together, we argue that the model can obtain good representations that are invariant to spatio-temporal transformations in a
time-varying input.

To test this hypothesis we perform an object recognition task in a video sequence. The goal here is to recognize the object in each frame given only four labeled frames per sequence. To test the performance, we use the COIL-100 [36] and Animal datasets [35], containing 100 and 60 different classes, respectively, and each class has a sequence of (roughly) 72 frames obtained by putting the object on a turn table. Fig. 7 shows some examples. We assume that the labels are available only for 4 frames, at $0^\circ, 90^\circ, 270^\circ$ and $360^\circ$ angles on the turn table, and the other orientations are used for testing. We assume similar experimental setup as is described in [37], where the representations inferred per each frame are considered as input features to a linear classifier. Table I shows the results obtained, where we compare sparse coding (SC) [38] with our dynamic sparse coding (DSC) and in addition, with invariant representations obtained from inferring the causes (DSC-I). We observe that dynamic sparse coding is able to perform better classification as it leverages the temporal context during inference. Further improvement can be achieved by using the causes that can encode the variations in the input better.

B. Recognizing Sequences in Noisy Data

As we discussed above, the top-down information helps to disambiguate the objects in a noisy environment by providing strong contextual priors to the lower levels. This helps the bottom layers extract information pertaining only to the object of interest, and pass it on to higher layers creating stable attractors of the responses unique for the object. In this section, using a simple experiment built from synthetic images, we show the role of these top-down connections during inference in the presence of structured noise. Video sequences consisting of two moving objects from the same type belonging to three different shapes (diamonds, triangles and squares - Refer to Fig. 8d) were constructed. The objective is to classify each video frame as coming from one of the three different classes. Several $32 \times 32$ pixel 100 frame long sequences were made using two objects of the same shape bouncing off each other and the “walls”. Many such sequences were then concatenated to form a 30,000 frames long video, which is used to train a two-layer network. Similar to the procedure described above, the bottom layer parameters, consisting of 100 dimensional states and 40 dimensional causes, are learned over a $20 \times 20$ pixel patch with a pooling of $2 \times 2$ and states having a receptive field of $12 \times 12$ pixels. Parameters of the second layer, consisting of 60 dimensional states and just 3 dimensional causes, are learned
Fig. 8: Shows the scatter plot of the 3 dimensional causes at the top-layer for (a) clean video with only bottom-up inference, (b) corrupted video with only bottom-up inference and (c) corrupted video with top-down flow along with bottom-up inference. At each point, the shape of the marker indicates the true shape of the object in the frame. (d) - (e) shows part of the clean and corrupted video sequences constructed using three different shapes. Each row indicates one sequence.

We test the performance of the model under two conditions. The first case uses 300 frames of clean video, with 100 frames per shape, concatenated without discontinuities as described above. In the second case, we corrupt the clean video with “structured” noise, where we randomly pick a number of objects from the same three shape classes with a Poisson distribution and add them to each frame independently at random locations. Fig. 8e shows an example when the mean of the Poisson distribution is 1.5 (SNR = 2.31)$^1$. There is no correlation between any two consecutive frames regarding where the “noisy objects” are added.

First, we consider the clean video and perform inference with only bottom-up information, i.e., during inference we consider $u^{(l)}_t = 0 \forall \ l \in \{1, 2\}$. Fig. 8a shows the scatter plot of the

$^1$We did not find any significant difference in the quality of the results when the mean is in the range of [1, 3] and SNR ranging between [3.6, -1.2].
three-dimensional causes at the top layer. Clearly, the model is able to separate the three shapes into three different clusters, representing a unique attractor for each object in the output space. Fig. 8b shows the scatter plot when the same procedure is applied on the noisy video. We observe that the three shapes cannot be clearly distinguished. Finally, we use top-down information, i.e., we use the cost function in (8) and make the top layer predicted causes equal to that of the previous frame ($\hat{u}^{L+1}_t = u^L_{t-1}$), along with the bottom-up inference as described above on the noisy data. As we argued before, since the second layer learned class specific information, the top-down information can help the bottom layer units to disambiguate the noisy objects from the true objects. Fig. 8c shows the scatter plot for this case. Clearly, with the top-down information, in spite of largely corrupted sequences, the model is able to separate the frames belonging to the three shapes (the connection from one cluster to the other is due to the temporal coherence imposed on the causes at the top layer). This shows that the contextual information coming from temporal and top-down connections disambiguates the noise and lets the model fall into a stable class attractor despite a large amount of corruption in the inputs.

VII. CONCLUSIONS

The proposed architecture for object recognition in video is an attempt to revisit the problem of extracting information from a complex time varying world using prior experience. Biological inspiration guided the design choices of a hierarchical, distributed, bidirectional architecture that learns in a self-organizing way parameters and features by explaining the input data. This is a daunting task, and therefore we must consider this an initial step that is testing basically three solutions. First, modeling should include dynamic components at the core because the world is time varying, and the state is a very effective way of implementing short-term memory as a signal that can be utilized readily for online processing. Second, the architecture should include bottom-up and top-down components to provide context at every level of processing. This bidirectional processing also allows for the use of past information to stabilize internal representations, and potentially the direct use of past information at the top of the pyramid to guide the extraction of information. Third, empirical Bayes provided the powerful algorithmic environment to solve this difficult problem, albeit with a huge computational complexity.

We provided an early analysis of the features obtained from natural video scenes with the proposed architecture. The type of basis, found by Olshausen and Fields [12] and mimicking V1
receptive fields, are also discovered in our approach from video scenes. Moreover, we also show that these bases smoothly vary in time, capturing the fact that objects move smoothly in natural scenes. The causes exploit local spatial redundancies, and are capable of combining the simpler representations of the bottom layer into increasingly more complex representations at the higher layers. Moreover, we also found that the feature dynamics slow towards the top, meaning that the top level representations are not only more complex but also more stable in time [28].

We tested a simple case of the use of external top-down information to disambiguate noisy video frames. In a synthetic video sequence we added correlated noise (i.e. the same objects) in random positions across time, which makes the task rather difficult. We introduced the top causes of the previous frame as the external cause of the current frame. This extra information was sufficient for the system to disregard the jumping noisy objects, and provide correct class identification. In the future, pre-stored gestalts in an associative memory can be used to bias the recognition system to pick special objects from cluttered scenes.

The cognitive architecture, as explained, does not scale up to large images without the use of specialized computer hardware tailored to the algorithms. Inference is a very computational demanding step because of the sparseness constraint, which is implemented as an iterative process, and requires high computational bandwidth. Having said this, it may be possible to implement sparsity constraints, not through L1 norms, but through nonlinearities in the generative model under (computationally efficient) parametric assumptions [13]. On the positive side, the approach is highly parallelizable and the locality of the data makes the use of GPUs an attractive alternative. The proposed architecture can also be combined with deconvolutional networks [33], which could also allow the model to scale to large images.

ACKNOWLEDGMENTS

This work is supported by the Office of Naval Research (ONR) grant #N000141010375.

REFERENCES


