Precision without Precisions: Handling uncertainty with a single predictive model

Benjamin Cowley[1], John Thornton[1,2], Linda Main[1] and Abdul Sattar[1]

[1]Institute for Integrated and Intelligent Systems, School of ICT, Griffith University, QLD 4111, Australia
[2]Department of Informatics, University of Sussex, Brighton BN1 9RH, UK
benjamin.cowley@griffithuni.edu.au

Abstract

The predictive processing theory of cognition and neural encoding dictates that hierarchical regions in the neocortex learn and encode predictive hypotheses of current and future stimuli. To better handle uncertainty these regions must also learn, infer, and encode the precision of stimuli. In this treatment we investigate the potential of handling uncertainty within a single learned predictive model. We exploit the rich predictive models formed by the learning of temporal sequences within a Hierarchical Temporal Memory (HTM) framework, a cortically inspired connectionist system of self-organizing predictive cells. We weight a cell's feedforward response by the degree of its own temporal expectations of a response. We test this model on data that has been saturated with temporal or spatial noise, and show significant improvements over traditional HTM systems. In addition we perform an experiment based on the Posner cuing task and show that the system displays phenomena consistent with attention and biased competition. We conclude that the observed effects are similar to those of precision encoding.

Introduction

Our senses are submersed in a sea of dynamic, complex, and noisy stimuli. To help in our navigation of the world Bayesian theories of cognition generally prescribe the need for sensory responses to have *precision weights* applied to them (Knill and Pouget, 2004). In this way sensory responses with high precision (inverse variance) will have a greater effect on our internal model's posterior probabilities than those with low precision. This approach is also adopted by the increasingly popular Bayesian theory of cognition and neural coding, *predictive processing* (Clark, 2013; Seth, 2014); also referred to as predictive coding (Rao and Ballard, 1999) or, when applied to Friston's generalization, free-energy minimization (Friston et al., 2006). In this treatment we will refer to it with the acronym PP.

In PP, hierarchically arranged regions encode predictive *hypotheses* of sensory information. Differences between an hypothesis and a stimulus are transmitted up the hierarchy by precision weighted error responses, which are then used to update the hypothesis, a process that continues iteratively and occurs end-to-end in the hierarchy. The hypotheses are encoded using a model learned with an objective function that minimizes surprisal, or a generalization thereof. Many contemporary treatments of PP also include the addition of *expected precisions*, which act in tandem with standard precision weighting (Hohwy, 2014). Expected precisions are projected top-down and represent the system's expectation of precisions given the current hypotheses. This allows the system to adjust precisions based on abstract cues (such as a pointing arrow) that may indicate a stimuli of high precision. Feldman and Friston (2010) applied such a model to a simulated version of Posner's cuing task (Posner, 1980) and showed that it produces attentional phenomena, such as increased neural responses and biased competition (Desimone and Duncan, 1995).

We argue that PP, as outlined above, requires the brain to adopt two seperate predictive models of the world, one that predicts stimuli and another that predicts the precision of stimuli. As a result PP requires two separate learning schemes: one that learns surprisal reducing hypotheses, and the other to learn the precision of responses under these hypotheses. Each region must also infer precisions from the current responses and encode them, while simultaneously encoding expected precisions, errors, and hypotheses.

In this treatment we investigate the potential of handling uncertainty within a single learned predictive model, applying only information derived from that model. Our approach exploits the temporal regularities in nature and perception to weight responses by the model's temporal expectations of a response, given both exogenous and endogenous causes. The weighting by exogenous causes allows the system to place greater trust in responses that conform to the rhythm and flow of the environment, while weighting by endogenous causes allows for more abstract and task specific context to guide the handling of uncertainty.

We implement our approach with the Hierarchical Temporal Memory (HTM) framework, a connectionist system based on the biology of the neocortex (Hawkins et al., 2010) with a number of similarities to PP, such as a hierarchical arrangement with both top-down and laterally projected hypotheses. For this particular study HTM provides a number

of advantages, the chief of which is the temporal sequence learning paradigm that enables an HTM system to form predictions on complex high level sequences at levels comparable to or better than other machine learning techniques (Cui et al., 2016). HTM produces this predictive performance while maintaining a high degree of biological plausibility, with self-organizing cells forming mono-directional synapses using Hebbian-based local learning rules (Hawkins and Ahmad, 2016). It is the aggregate activity of these cells that allows regions to learn, encode, and predict on sequences, while the aggregate of these regions offers tantalizing explanations for our cognitive experience.

Our approach uses the state information of the multitude of cells in an HTM region to form inferences on the uncertainty of stimuli, and then weight the feedforward responses of each cell. We integrate this into a fully hierarchical HTM system, which we refer to as Hierarchical Temporal Memory with Predictive Bias (HTM-PB). As most published HTM work focuses on single region models, our work also helps to fill this gap in the literature.

To ascertain HTM-PB's effectiveness given noisy stimuli we test the system on high-order sequence data that has been injected with temporal and spatial noise. Our results indicate that the HTM-PB can improve HTM's performance on noisy data. Our second round of experiments are designed to test whether HTM-PB can produce phenomena consistent with attention, like expected precisions in PP. Following Feldman and Friston's (2010) previous work with PP and precisions, we apply the system to a dataset based on the Posner cuing task (Posner, 1980). Here HTM-PB's responses show similar characteristics to Posner's human trials. We end with a discussion on the possible implications of this work regarding HTM, PP, and cognition and neural encoding in general.

Hierarchical Temporal Memory

HTM is directly based on observations of neuronal function in the neocortex (Hawkins and Ahmad, 2016). It models neocortical pyramidal cells as organized into functional columns, such that pyramidal cells that occupy the same column share the same receptive field on their proximal dendrites (dendritic branches closest to the soma). Each cell's distal dendrites (branches further from the soma) self-organize to learn temporal causes of activity on the proximal dendrites. This is achieved using a Hebbian learning approach that guides the forming of synapses to other cells. Activity on the distal dendrites constitute a "prediction" of activity on the proximal dendrites which generates depolarization of the cell. If a cell then receives activity on its proximal dendrites, the depolarization caused by the prediction leads the cell to produce a strong response which inhibits other cells in the column. Apical dendrites (a long branch extending from the top of the cell) perform a similar role to distal dendrites, but instead form synapses to axons that extend from regions higher in the cortical hierarchy.

Below we provide a brief overview of our implementation of HTM, followed by an algorithmic formalization for encoding the feedforward and feedback outputs of a region (we detail the feedforward output for HTM-PB in a later section). For a more detailed explanation of the algorithms we recommend the Cortical Learning Algorithms white paper (Hawkins et al., 2010); for the specific version of the spatial pooler we use, augmented spatial pooler, we recommend Thornton and Srbic (2013).

Implementation

HTM implementations model neuronal activity as discrete states. The column states (i.e. the proximal dendrite activity) are computed by the spatial pooler (SP) algorithm, while the states of individual cells are computed by the temporal pooler (TP).

The SP handles boolean input, however, as we intend using real valued input for predictive bias we opt to use the augmented spatial pooler (ASP), which is an extension of SP that allows for real valued input (Thornton and Srbic, 2013). ASP also has a convergence function that allows for offline training by iterating over the input until stable representations are found, a feature we also employ in our experiments. Although SP and ASP differ in these regards, the primary processes of computing the input and performing *inhibition* remain the same. Each column has a number of connected synapses to the feedforward input that are learned using a Hebbian learning scheme. On each timestep, a column's activation level, termed the *overlap*, is calculated as the sum of the feedforward input received on that column's connected synapses. Columns with the highest overlap inhibit neighbouring columns, placing them in an *inactive state*, while uninhibited columns are in an *active state*.

The TP algorithm models distal dendrites and drives the sequence prediction of HTM. Each cell has a number of dendrite *segments*, which can be one of two types: *sequence segments*, which predict the next timestep in a sequence; and *non-sequence segments*, which predict timesteps following the next. If one of these segments becomes active then the cell enters a *predictive state*; if on the next timestep the column becomes active then only those cells that were in a predictive state become active. If it was not predicted then all cells in the column become active. When this happens a cell of that column will form synapses, on a sequence segment, to cells that were active on the preceding timestep. On any given timestep, if the number of synapses connected to active cells on a segment is greater than an activation threshold, then that cell enters a *predictive state*. Non-sequence segments differ only in that they form synapses to cells that were active in the timestep preceding a correct prediction by another segment. In this way non-sequence segments form predictions on predictions, and allow a cell to enter a predictive state several timesteps prior to expected activity.

Most research into HTM algorithms have focused on sin-

gle region implementations, as such there is no canonical implementation of an HTM hierarchy. Our approach is derived from Kneller and Thornton's (2015) work. However, that study was applied to static images and did not use the TP, so only activity was communicated between regions, whereas here we also communicate predictions. Each column of a region has a feedback *axon* that projects down to lower regions. If a column is active or if one of its cells is in a predictive state from a sequence segment, then the state variable for the column's feedback axon is active. Cells in a lower region connect to these axons using the same learning method as sequence segments in TP. Each cell has a feedforward output axon and this is active if the cell is active or in a predictive state. The input into an ASP from a higher region is the feedforward output from a lower region.

Encoding Input and Output of a Region

ASP: We store the overlap of each column an array, $\mathbf{o}$, of length equal to the number of columns in the region, n. Given an input array of real values, $\mathbf{x}'$, of length m' we can then calculate the overlap score for a column, j, using the function:

$$\text{overlap}(\mathbf{x}', \mathbf{S}, j) = \sum_{s \in \mathbf{S}_{(j,\dots)}} \mathbf{x}'_{(s)}$$

where $\mathbf{S}$ is a two dimensional array of size $n \times m'$ that contains the index of each connected synapse s. Inhibition can be performed sequentially through the columns by ranking adjacent columns according to their activation levels and inhibiting those which rank below the desired activity level for sparsification, γ (a real value indicating the ratio of activity to columns). To determine how many adjacent columns are used during inhibition, a common receptive field size is first calculated as the mean receptive field size, μ. All other columns that are within this distance are included in the same neighbourhood as the column (see Hawkins et al. (2010) for more detail). We store the neighbourhoods in a two dimensional array, $\mathbf{C}$, of size $n \times \mu$. We calculate the inhibition for a column, j, using Algorithm 1, where b is a count of the columns with a greater overlap within the neighborhood of column j; and $\hat{j}$ is the index of $\mathbf{C}_{(j,\dots)}$ for each neighboring column of j. We store the final sparsified activity for all columns in a boolean array, $\mathbf{y}$, of length n.

TP: We store the predictive state variable of all cells in a boolean array, $\boldsymbol{\pi}$, with length mn, where m is the number of cells in each column. To calculate a cell's activity we take $\mathbf{y}$, as calculated above, and the index of the cell we wish to calculate, i, and apply the function:

$$\text{act}(\mathbf{y}, \boldsymbol{\pi}, i, j, m) = \begin{cases} 1 & \text{if } \mathbf{y}_{(j)} = 1 \wedge \big(\boldsymbol{\pi}_{(i)} = 1 \vee \\ & \quad \forall \pi \in \boldsymbol{\pi}_{(jm+1 \dots jm+m)} (\pi = 0)\big) \\ 0 & \text{otherwise} \end{cases}$$

Algorithm 1 inhibition($\mathbf{o}, j, \mathbf{C}, \mu, \gamma$)

```
b = 0
for ĵ ← 1 to μ do
    c = C_(j,ĵ)
    if o_c > o_j then
        b ← b + 1
if b/μ < γ then
    return 1
else
    return 0
```

which requires that $1 \leq i \leq mn$. Using this function we build a boolean array, $\mathbf{a}$, of length m containing the active states of cells. Next we calculate the predictive states using the function:

$$\text{predict}(\mathbf{a}, \mathbf{D}, i, q, r, \theta) = \begin{cases} 1 & \text{if } \exists \mathbf{d} \in \mathbf{D}_{(i,1\dots q,\dots)} \\ & \quad \left(\left(\sum_{d \in \mathbf{d}} [\mathbf{a}_{(d)} = 1] \right) \geq \theta \right) \\ 0 & \text{otherwise} \end{cases}$$

where q is the maximum number of segments and r is the maximum number of synapses; $\mathbf{D}$ is a multi dimensional array of size $mn \times q \times r$ where each entry is the index of a cell with which the current cell, i, has a connected synapse; θ is the activation threshold. So a cell will be in a predictive state if one of its segments has a number of connected synapses to active cells that is greater than or equal to θ. As can be seen in Algorithm 2, we use the same function for sequence, non-sequence, and feedback segments. To do this we simply pass different arrays of synapses: $\dot{\mathbf{D}}$ (for sequence seqments), $\ddot{\mathbf{D}}$ (for non-sequence segments), and $\breve{\mathbf{D}}$ (for feedback segments). In the case of forming predictions on feedback we pass in the feedback activity, $\boldsymbol{\alpha}'$, instead of $\mathbf{a}$. We store the predictive states in arrays $\dot{\mathbf{I}}$ (for cells predicted to be active next), $\ddot{\mathbf{I}}$ (for cells predicted to be active in the future), and $\breve{\mathbf{I}}$ (for cells predicted by feedback). These arrays are of size $\hat{t} \times mn$, where $\hat{t}$ is the number of timesteps we wish to record. Given these arrays we can calculate $\boldsymbol{\pi}$ (as used in the act function) as an element-wise logical or (inclusive disjunction) of $\dot{\mathbf{I}}$, $\ddot{\mathbf{I}}$, and $\breve{\mathbf{I}}$. To calculate the feedforward output of a region we perform a logical or on the predictive and active states of each cell.

Boolean Hierarchy: We store a region's feedback output in a boolean array, $\boldsymbol{\alpha}$, of length n. In lines 15 to 19 of Algorithm 2, we calculate each each entry, j, such that it will have a value of 1 if column j is active or if one the column's cells is predicted by a sequence segment. To calculate the feedforward output of a region we perform a logical or on the predictive and active states of each cell, as seen in line 19 of Algorithm 2. We store the feedforward output in an array, $\mathbf{x}$, of length m. The $\mathbf{x}$ array will be the feedforward

input, $\mathbf{x}'$, for the region above it in the hierarchy, while $\boldsymbol{\alpha}$ will be the feedback input, $\boldsymbol{\alpha}'$, for the region below it.

Algorithm 2 regionEncode

Input: b // If $b = 1$ then we use predictive bias
Input: $\mathbf{x}', \boldsymbol{\alpha}'$ // Input from other regions
Input: $\dot{\mathbf{I}}, \ddot{\mathbf{I}}, \breve{\mathbf{I}}$ // Predictions from previous timesteps
Input: $\mathbf{S}, \dot{\mathbf{D}}, \ddot{\mathbf{D}}, \breve{\mathbf{D}}, \mathbf{C}, \mu$ // Learned structures
Input: $t, m, n, q, r, \breve{q}, \breve{r}, \gamma, \theta, \breve{\theta}$ // Parameters

for $j \leftarrow 1$ **to** n **do**
  $\mathbf{o}_{(j)} \leftarrow \text{overlap}(\mathbf{x}', \mathbf{S}, j)$
for $j \leftarrow 1$ **to** n **do**
  $\mathbf{y}_{(j)} \leftarrow \text{inhibit}(\mathbf{o}, j, \mathbf{C}, \mu, \gamma)$
for $i \leftarrow 1$ **to** mn **do**
  if $\dot{\mathbf{I}}_{(t-1,i)} \vee \ddot{\mathbf{I}}_{(t-1,i)} \vee \breve{\mathbf{I}}_{(t-1,i)}$ **then**
    $\boldsymbol{\pi}_{(i)} \leftarrow 1$
  else
    $\boldsymbol{\pi}_{(i)} \leftarrow 0$
  $\mathbf{a}_{(i)} = \text{act}(\mathbf{y}, \boldsymbol{\pi}, i, \lfloor \frac{i}{m} \rfloor, m)$
for $i \leftarrow 1$ **to** mn **do**
  $\dot{\mathbf{I}}_{(t,i)} = \text{predict}(\mathbf{a}, \dot{\mathbf{D}}, i, q, r, \theta)$
  $\ddot{\mathbf{I}}_{(t,i)} = \text{predict}(\mathbf{a}, \ddot{\mathbf{D}}, i, q, r, \theta)$
  $\breve{\mathbf{I}}_{(t,i)} = \text{predict}(\boldsymbol{\alpha}', \breve{\mathbf{D}}, i, \breve{q}, \breve{r}, \breve{\theta})$
for $j \leftarrow 1$ **to** m **do**
  $\boldsymbol{\alpha}_{(j)} \leftarrow 0$
  for $i \leftarrow jn + 1$ **to** $jn + n$ **do**
    if $(\mathbf{a}_{(j)} \vee \dot{\mathbf{I}}_{(t,i)}) = 1$ **then**
      $\boldsymbol{\alpha}_{(j)} \leftarrow 1$
for i $\leftarrow 1$ **to** mn **do**
  if $b = 0$ **then**
    $\mathbf{x}_{(i)} \leftarrow \mathbf{a}_{(i)} \vee \dot{\mathbf{I}}_{(t,i)} \vee \ddot{\mathbf{I}}_{(t,i)} \vee \breve{\mathbf{I}}_{(t,i)}$
  else
    $\mathbf{x}_{(i)} \leftarrow \text{encodePredictiveBias}(\mathbf{a}, \dot{\mathbf{I}}, \ddot{\mathbf{I}}, \breve{\mathbf{I}}, i, t)$
return $\mathbf{x}, \boldsymbol{\alpha}$

HTM with Predictive Bias

Standard HTM, like more typical artificial neural network implementations, provides no explicit method for handling uncertainty of input, instead it relies on the robustness of the system. From a Bayesian perspective all responses are, essentially, assigned a precision of 1.0 and have equal say in the forming of hypotheses. With our predictive bias method, a cell's response is weighted by its temporal expectations. Those responses with greater expectations will have a greater effect on the higher-level encoding that is produced by ASP's competitive inhibition scheme. Given that this treatment is a theoretical investigation, our method is not overly optimized, instead our aim is that the method be simple and intuitive.

With this in mind, we use a simple additive scoring system for HTM-PB as outlined in Algorithm 3, where each cell is initially assigned a score, s, of 0.0. If a cell is active we score the cell based on its predictive states in the previous timestep; a cell that is active will have a greater score if it was predicted, and a lower score if it was not. We also score an inactive cell in the same way if it was predicted by a sequence segment in the previous timestep. Using this method a cell that was not predicted but is active may have a lower score than a cell that was predicted but was not active. This is an important addition, as we are putting far less trust in the quality of the input than standard HTM implementations. We also weight cells according to their predictive states, if they were not predicted by a sequence segment on the previous timestep.

As can be seen in Algorithm 3 we add a score of 0.25 if the cell is active. If the cell was predicted by a sequence segment we also increment the score by 0.25. This takes into account the exogenous causes, as the input of the previous timestep directly informs our expectations for the next. We also increment the score by 0.25 given predictions formed by feedback segments. This allows for endogenous, task dependent causes, to modulate responses. Next, we take into account how long the cell was predicted to be active by adding a score of 0.05 for each timestep that a cell has been in a predictive state by the lateral connections ($\dot{\mathbf{I}}$, $\ddot{\mathbf{I}}$), to a maximum of 0.25. So the greater the number of accumulated causes for a response the greater the trust we place in a response. The maximum score is 1.0 which only occurs if a cell is active and was "perfectly" predicted.

Algorithm 3 encodePredictiveBias($\mathbf{A}, \dot{\mathbf{I}}, \ddot{\mathbf{I}}, \breve{\mathbf{I}}, i, m, t$)

$s \leftarrow 0.0$
if $\mathbf{a}_{(i)} = 1 \vee \dot{\mathbf{I}}_{(t-1,i)} = 1$ **then**
  $\breve{t} \leftarrow t - 1$
else
  $\breve{t} \leftarrow t$
if $\mathbf{a}_{(i)} = 1$ **then**
  $s \leftarrow s + 0.25$
if $\dot{\mathbf{I}}_{(\breve{t},i)} = 1$ **then**
  $s \leftarrow s + 0.25$
if $\breve{\mathbf{I}}_{(\breve{t},i)} = 1$ **then**
  $s \leftarrow s + 0.25$
$\acute{s} \leftarrow 0.0$
$\acute{t} \leftarrow \breve{t}$
while $(\dot{\mathbf{I}}_{(\acute{t},i)} = 1 \vee \ddot{\mathbf{I}}_{(\acute{t},i)} = 1) \wedge \acute{s} < 0.25$ **do**
  $\acute{s} \leftarrow \acute{s} + 0.05$
  $\acute{t} \leftarrow \acute{t} - 1$
$s \leftarrow s + \acute{s}$
return s

Experiments and Results

We perform two sets of experiments to test HTM-PB. In the first set we use a dataset with varying degrees of spatial and temporal noise to ascertain whether HTM-PB im-

proves HTM's robustness to noise. This is important as precision weighting in PP is also intended to improve predictive performance on noisy stimuli. Our second experiment tests HTM-PB in relation to Posner's cuing task. The aim is to ascertain if HTM-PB displays phenomena consistent with attention, as expected precisions in PP do.

Temporal and Spatial Noise

In these experiments we apply the same experimental setup as Cui et al. (2016) when testing a single region of the temporal pooler. In their experiments they used a single TP region that did not have non-sequence segments, so it only predicted the next timestep; a TP utilizing this set up is referred to as temporal memory (TM). Here we compare the results of HTM-PB to a boolean hierarchy (HTM-B), a TP with sequence segments, and a TM.

Dataset: The dataset is 16,000 timesteps in length and comprises eight sequences of length seven or eight, that appear randomly. The sequences do not overlap and are separated by a single timestep of noise. After training for 12,000 timesteps, a noise element replaces either timestep 2, 3, or 4 of every sequence for the remainder of the dataset. This is a difficult problem, as the sequences share many subsequences and to correctly predict all timesteps a high-order prediction system is required. However the noise can disrupt a prediction system's trajectory and it 'forgets' the earlier elements.

In addition to this we also test the systems with no noise, with two and three consecutive timesteps of noise, and also spatial noise on all timesteps within a sequence. In the spatial domain we apply noise levels ranging from 10% to 50% (all noise is added after 12,000 timesteps). For each active entry of the input array we *flip* it to inactive with a probability given by the level of noise, so with 50% noise level an active bit will have a probability of 0.5 of being inactive. However, we assume the sparse encoding strategy is still intact, so we don't use the same values to flip the inactive elements (otherwise 50% of elements will be active). Instead we multiply the probability of being flipped by the probability of an element being active under the sparse coding strategy, which is 0.02 for this dataset. So with noise of 50% an inactive element will have a probability of being flipped to active of 0.01.

Experimental Setup and Metrics: Following Cui et al. we do not use a spatial pooler on the input, instead feeding it directly into the TP. We use the same parameters as Cui et al. for TM but for the other algorithms we use less cells per column (10 instead of 32), to promote computational efficiency in the hierarchy. Both hierarchies have two stacked regions and the second (higher) region has 1024 columns, which are activated by an ASP with a γ of 0.02. Cells in the second layer TP and the feedback segments in the first region have a θ and r of 8 and 16 respectively (lower regions and TM have 15 and 128). To train the hierarchy we run the first regions' TP over the input for 5,000 timesteps, and then turn learning off and train the higher regions' ASP on these codes until convergence. We then run the top regions' TP over these codes, switch learning back on, and run the remainder of the timesteps with the full hierarchy.

On the penultimate timestep of a sequence we take the predictions produced by sequence segments and use a nearest neighbour classification approach to predict the final timestep. Of all the possible sparse codes in the dataset (5011 in total) the one that has the greatest number of matches to predictions is classified as the next sparse code (ties are broken randomly). The accuracy is given as the ratio of correctly classified final codes. We perform 10 runs for each system using a different random seed (1 to 10) on each dataset, and provide the averaged results.

	TM	TP	HTM-B	HTM-PB
No noise	**1.0000**	**1.0000**	0.9976	**1.0000**
Temporal 1	0.4830	0.4660	0.7000	**0.9113**
Temporal 2	0.4830	0.4596	0.4801	**0.7106**
Temporal 3	**0.4830**	0.4319	0.4617	0.4603
Spatial 10%	**1.0000**	**1.0000**	0.9975	**1.0000**
Spatial 20%	**1.0000**	**1.0000**	0.9970	**1.0000**
Spatial 30%	**1.0000**	**1.0000**	0.9813	**1.0000**
Spatial 40%	0.8945	0.9411	0.9092	**0.9967**
Spatial 50%	0.2068	0.2823	0.3527	**0.4123**

Table 1: Results, in terms of accuracy, for the experiments with no noise, temporal noise, and spatial noise.

Results and Analysis: As can be seen from the results in Table 1 the algorithms perform very well with no noise (with only HTM-B not achieving perfect results). However, when temporal noise is added the single region systems' performance drops considerably. With one timestep of noise HTM-PB's accuracy remains over 0.9, HTM-B also performs better than single regions here. This indicates that the addition of a hierarchy can improve accuracy on HTM regions for single timesteps of noise, while predictive bias provides greater improvement and can maintain higher accuracy on two timesteps of noise. For spatial accuracy all HTM systems show remarkable robustness up to 40% noise, before dropping notably. HTM-PB attains the highest accuracy on all spatial noise levels, with near perfect accuracy at 40% and double that of TM's at 50%.

These results indicate that HTM-PB improves the accuracy of HTM systems on noisy data. The higher accuracy levels for HTM-PB on temporal noise 1 and 2, as well as spatial noise 40% and 50% are statistically significant, using an ANOVA with a 95% confidence interval. The TM's improved performance on temporal noise 3 is also statistically significant. TM's moderately improved performance

there is, we suspect, caused by its having more cells per column. HTM-B's performance is often below that of single regions, which indicates that the hierarchical models may require additional training to reach peak performance.

Posner Cuing Task

The original Posner cuing task was designed to investigate the interactions of covert visual attention (no eye movement) and reaction time (Posner, 1980). Human subjects were tasked with pressing a button when a stimulus (luminance increment) was presented. This stimulus could appear to either the left or right of the subjects' gaze, which was fixated on a central position. A *cue* would appear that indicates where the stimulus would be likely to appear. If the cue was a plus sign the stimulus would have a 0.5 probability of appearing on either the left or right. If the cue was an arrow the stimulus would have a 0.8 probability of appearing in the direction that the arrow pointed. The trials were split into three categories: valid, where the stimulus appears on the side that an arrow cue points; invalid, where the stimulus appears on the side that an arrow cue does not point; and neutral, where the cue is a plus and the stimulus appears on either side. The results of these experiments showed that covert attention directed by cues affects reaction time, improving it where cues direct (valid trials), and significantly reducing it where it counters the cue's direction (invalid trials). These results can be effectually displayed in graph form, which we have reproduced in Figure 2a.

In this experiment we test HTM-PB to see if it produces similar phenomena. We assume that for recognition and reaction to occur, the stimulus must be represented in levels of the hierarchy which are higher than the initial perceptual region. Therefore, in these experiments we focus on the responses of the top region of the hierarchy.

Dataset: We crafted a dataset directly based on the Posner cuing task, with black and white images of size 56×14 pixels depicting cues and stimuli, examples of which are provided in Figure 1. Each trial within the dataset is 20 timesteps long; five timesteps of just the cue, followed by 15 timesteps of the cue and stimulus. There are three different cues: plus (neutral), left arrow, and right arrow. A stimulus can appear on either side of a cue using the same probabilities as the original experiment (as listed above). Each trial is followed by 10 timesteps of random noise that separates the trials and also helps the ASP converge (as convergence is difficult with only a small number of images). The dataset contains 200 separate trials, for a total of 6000 timesteps.

Experimental Setup and Metrics: For this experiment we use a much slower learning rate, with a synapse increment of 0.01, and a decrement of 0.002. We found this necessary due to the simplicity of the sequences, which we discuss further in the analysis. We again use a two region

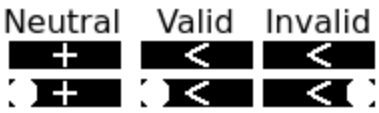

Figure 1: Example input images for the Posner cuing task.

hierarchy for HTM-PB, but due to the different input size we use a different number of columns: the first region has 784, the second has 392. As we have less columns, we use a smaller activation threshold: 8 for the first region, and 5 for the second. Exploratory experiments showed ASP encodes images slightly better with decreased sparsity, so we use a desired local activity of 0.04. For training, we run the first region ASP to convergence on the dataset. Over these codes we train the first region TP for one iteration over the dataset, and then a second iteration with learning off. We train the second region ASP to convergence on the output of the first region's second iteration, and then train the second region TP on these codes. We then enable feedback and perform another full iteration of the dataset. We then run the final trials and extract the results.

The sequences we use for the trials are very simple: five timesteps of a static input, followed by 15 timesteps of another static input. Because of this we expect the top level to reach a stable encoding (i.e. no change in states between timesteps) before the end of the sequence. We assume that this stable encoding is the best possible encoding that the region can produce of the input. As a metric we measure the average number of timesteps taken for the top region of the HTM-PB to reach this final encoding. These metrics are clearly different to the original Posner Cuing Task experiments which used the average time for participants to press button in response to the stimulus. As the inclusion of motor commands in the HTM framework lies beyond the scope of this treatment, we assume that the closer the encoding is to the best possible encoding, the more likely it is that a recognition and accompanying reaction would occur. We also diverge from Feldman and Friston's metrics which measured the responses of the artificial neurons that were dedicated to encoding either the left or right stimulus. Our model has no such dedicated neurons, instead connections are entirely learned. The final results are averaged over 30 runs, each using a different random seed (1 to 30).

Results and Analysis: Figure 2b shows the mean number of timesteps to reach the final encoding of the input for invalid, neutral, and valid trials. As can be seen, these results match the proportionality of the human trials surprisingly closely. However, there is a lot of variance in these results, as indicated by the error bars. The difference between valid and invalid is significant, however neutral trials show no statistically significant difference with valid or invalid over the

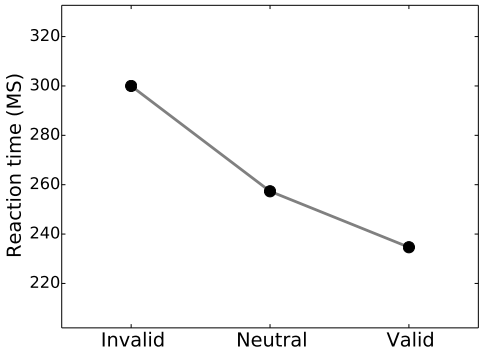

Figure 2a: Results of the original Posner cuing task on human subjects, taken from Posner (1980).

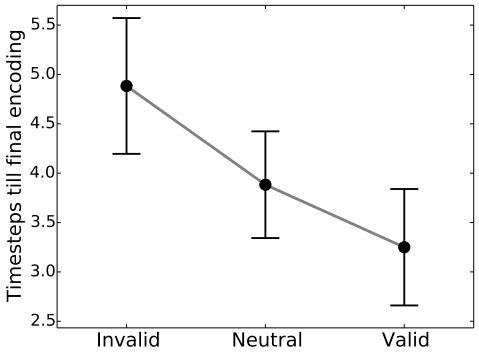

Figure 2b: Results from our HTM Posner cuing task experiment, with 95% confidence interval error bars.

30 runs of the experiment.

The results above were achieved with only minor changes to the system parameters used in the first experiments, which were on very different datasets and domains. The major change was to slow the learning rate on the Posner cuing task. The fast learning rate used in the first experiments helps to quickly learn high order sequences, however it can cause it to overfit recent inputs. This causes the system to be biased to last seen trials in Posner cuing task (e.g. if the last trial with a left pointing cue was invalid, the next time it sees a left pointing cue it predicts an invalid trial). This is easily remedied by lowering the learning rate so that system learns long term regularities and not just the recent past.

Biased Competition: Biased competition is a theory that stimuli "compete" to be represented in neural encodings, and stimuli that are the target of attention enjoy a bias in this competition. Thus, neural responses to an attended stimulus are greater than those of an unattended stimulus. Following Feldman and Friston (2010) we perform an additional experiment in which we present both valid and invalid stimuli simultaneously following a cue. If there is bias competition taking place we would expect that the cells encoding the valid stimulus for the cue would respond more strongly than those that encode the invalid cue.

We display the results for this trial in Figure 3, which were taken from the timestep the stimuli were first presented (timestep 5). The images were made by tracing the connections from active columns in the top layer, down through the bottom region to the individual entries of the input. For each entry connected in this way we increment the value of the corresponding pixel. This is then normalized, such that the highest value is 1.0, and visualized as a grayscale image.

As can be seen in Figure 3, the columns which encode the valid stimulus are far more active than those that encode the invalid stimulus, even though both stimuli are presented on the same timestep and with the same input intensities. This indicates that the predictive bias is working as expected, and the cells that encode the predicted stimulus (valid) are responding more strongly. This is caused by the columns of the second region having larger overlap scores (due to the predictive bias scheme) which inhibits the columns with synapses to the cells that encode the invalid stimulus. As a result, the responses from the invalid stimulus encoding columns are much weaker on this trial than during the invalid trials. In fact they are only slightly stronger than their responses during the valid trials. It is interesting that there are responses from these columns at all when the valid stimulus is displayed during valid trials. This phenomena is a result of HTM-PB's encodings being probabilistic representations and given a left pointing arrow there is still a (small) probability that a stimulus will appear on the right.

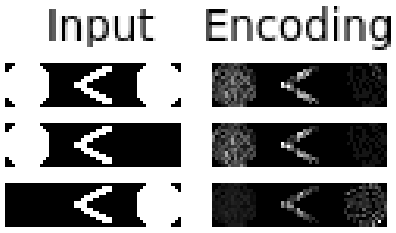

Figure 3: Visualization of bias competition experiment.

Discussion and Conclusions

In this study we set out to emulate the effects of precision and expected precision weighting solely through the HTM's existing predictive capabilities. To do this we modulated the feedforward response of each cell based on a scoring scheme that exploited both the sequence learning and hierarchical paradigms inherent in an HTM system. The effect of these paradigms allows each cell within a region to learn multiple causes for their activity. Under our method a cell's response will be greater if it has accumulated more causes indicating that it should respond. Our motivation was that, if an HTM

system had learned a rich model, the number of accumulated causes for activity may act as a measure of certainty.

Our experiments with both noisy data and the Posner cuing task indicate that our predictive bias scheme does reproduce the effects we would expect with precisions. So, assuming the neocortex does indeed adopt a hierarchical sequence learning strategy, it may not require the direct estimation and encoding of precisions to handle uncertainty. Instead it could rely on a rich predictive model based on temporal regularities in the environment, which allow it to infer the uncertainty of stimulus based on the stimulus' conformation to the system's model of the world. A predictive model with this form of uncertainty handling adopts a greater role for prediction in cognition, with hypotheses of the world even further biased by our expectations of how the world should behave. With regards to attention, we simply attend to what we predict to attend. This recasts the attentional task of searching for an object as being achieved by an endogenous prediction that we will perceive the stimuli that is associated with the object. This process would cause cells that encode the stimuli associated with the object to produce stronger responses and have a greater impact on the higher level encoding due to biased competition. We believe this is a simpler approach than PP's less direct role for predictions: where the stimuli will produce greater responses because we have learned to associate the stimuli as having high precision when we are predicting to perceive the object.

An obvious counterpoint to our recasting of attention is that, during free viewing tasks, saccades (eye movements) target content that exhibits a high difference between prior and posterior expectations, a phenomenon often referred to as *Bayesian surprise* (Itti and Baldi, 2005). This problem is also encountered by typical PP approaches, and is related to the dark room problem: if our goal is to minimize surprise, why not just retire to a dark room? Friston et al. (2012) demonstrated that this problem can be avoided in PP if we apply prior expectations that we will be surprised. These prior expectations allow an active inference approach to saccadic control to target novelty, enabling the building of hypotheses that better explain the novelty. It should be noted that in this model saccades were driven, not by expected precisions, but by active inference (which allows motor control to be modelled as a form of inference). With this in mind, we argue that for HTM-PB to produce Bayesian surprise phenomena we require an HTM system that moves away from the passive perceptive system, as it currently stands, and to equip it with active inference capabilities.

We also showed that HTM-PB's strategy for handling uncertainty can significantly improve the HTM framework's performance on noisy data. This improvement is over both a single region HTM and an HTM hierarchy. These results suggest that implementations of HTM hierarchies should adopt a predictive bias scheme for encoding feedforward input. As the goal of this work was a theoretical proof of concept, the method we used to modulate the cells response applied a simple additive scheme. Future work could look into possible optimizations of this method as well as applications of HTM-PB for industry. A more theoretical goal for future work would be the adoption of active inference and motor control within an HTM system; coupled with predictive bias, this would further advance HTM's scope as a cognitive system and open up new fields for applications.

References

Clark, A. (2013). Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and brain sciences*, 36(3):181–204.

Cui, Y., Ahmad, S., and Hawkins, J. (2016). Continuous online sequence learning with an unsupervised neural network model. *Neural computation*, 28(11):2474–2504.

Desimone, R. and Duncan, J. (1995). Neural mechanisms of selective visual attention. *Annual review of neuroscience*, 18(1):193–222.

Feldman, H. and Friston, K. (2010). Attention, uncertainty, and free-energy. *Frontiers in human neuroscience*, 4:215.

Friston, K., Adams, R., Perrinet, L., and Breakspear, M. (2012). Perceptions as hypotheses: saccades as experiments. *Frontiers in psychology*, 3:151.

Friston, K., Kilner, J., and Harrison, L. (2006). A free energy principle for the brain. *Journal of physiology-Paris*, 100(1):70–87.

Hawkins, J. and Ahmad, S. (2016). Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in neural circuits*, 10:23.

Hawkins, J., Ahmad, S., and Dubinsky, D. (2010). Hierarchical temporal memory including HTM cortical learning algorithms. *Techical report, Numenta, Inc, Palto Alto*.

Hohwy, J. (2014). *The neural organ explains the mind*. Open MIND. Frankfurt am Main: MIND Group.

Itti, L. and Baldi, P. F. (2005). Bayesian surprise attracts human attention. In *Advances in neural information processing systems*, pages 547–554.

Kneller, A. and Thornton, J. (2015). Distal dendrite feedback in hierarchical temporal memory. In *International joint conference on neural networks (IJCNN 2015)*. IEEE.

Knill, D. C. and Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *Trends in neurosciences*, 27(12):712–719.

Posner, M. I. (1980). Orienting of attention. *Quarterly journal of experimental psychology*, 32(1):3–25.

Rao, R. P. and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87.

Seth, A. K. (2014). The cybernetic Bayesian brain. In *Open Mind*. Open MIND. Frankfurt am Main: MIND Group.

Thornton, J. and Srbic, A. (2013). Spatial pooling for greyscale images. *International journal of machine learning and cybernetics*, 4(3):207–216.