Title
Active Predictive Coding: A Unified Neural Framework for Learning Hierarchical World Models for Perception and Planning

Journal
Proceedings of the Annual Meeting of the Cognitive Science Society, 45(45)

Authors
Rao, Rajesh P. N.
Gklezakos, Dimitrios Christoforos
Sathish, Vishwas

Publication Date
2023

Peer reviewed

Active Predictive Coding: A Unified Neural Framework for Learning Hierarchical World Models for Perception and Planning

Rajesh P. N. Rao, Dimitrios C. Gklezakos, Vishwas Sathish
{rao,gklezd,vsathish}@cs.washington.edu
Center for Neurotechnology and
Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle

Abstract

Predictive coding has emerged as a prominent model of how the brain learns through predictions and prediction errors. Traditional predictive coding focused primarily on sensory coding and perception. Here we propose active predictive coding (APC), a unified framework for perception, action and cognition. By learning hierarchical world models, the APC framework addresses important open problems in cognitive science and AI such as: (1) how do we learn compositional representations, e.g., part-whole hierarchies for equivariant vision? and (2) how do we solve large-scale planning problems, which are hard for traditional reinforcement learning, by composing complex action sequences from primitive policies? APC exploits hypernetworks, self-supervised learning and reinforcement learning to learn hierarchical models that combine task-invariant state transition networks and task-dependent policy networks at multiple abstraction levels. We illustrate the applicability of the APC model to active visual perception and hierarchical planning. Our results represent, to our knowledge, the first proof-of-concept demonstration of a unified approach to addressing the part-whole learning problem in vision, the nested reference frames learning problem in cognition, and the integrated state-action hierarchy learning problem in reinforcement learning.

Keywords: Predictive coding; neural modeling; vision; hierarchical reinforcement learning

Introduction

Predictive coding (Rao & Ballard, 1999; Friston & Kiebel, 2009; Keller & Mrsic-Flogel, 2018; Jiang & Rao, 2022b) has received increasing attention in recent years as a model of how the brain learns models of the world through prediction and self-supervised learning. In predictive coding, feedback connections from a higher to a lower level of a cortical neural network (e.g., the visual cortex) convey predictions of lower-level responses and the prediction errors are conveyed via feedforward connections to correct the higher-level estimates, completing a prediction-error-correction cycle. Such a model has provided explanations for a wide variety of neural and cognitive phenomena (Keller & Mrsic-Flogel, 2018; Jiang & Rao, 2022b). The layered architecture of the cortex is remarkably similar across cortical areas (Mountcastle, 1978), hinting at a common computational principle, with superficial layers receiving and processing sensory information and deeper layers conveying outputs to motor centers (Sherman & Guillery, 2013). The traditional predictive coding model focused on learning visual hierarchical representations and did not acknowledge the important role of actions in learning world models.

In this paper, we introduce Active Predictive Coding (APC), a new model of predictive coding that combines state and action networks at different abstraction levels to learn hierarchical internal models. The model provides a unified framework for addressing several important but seemingly unrelated problems in perception, action, and cognition as described below.

Part-Whole Learning Problem. Hinton and colleagues have posed the problem of how neural networks can learn to parse visual scenes into part-whole hierarchies by dynamically allocating nodes in a parse tree. They have explored networks that use a group of neurons to represent not only the presence of an object but also parameters such as position and orientation (Sabour, Frosst, & Hinton, 2017; Kosiorek, Sabour, Teh, & Hinton, 2019; Hinton, Sabour, & Frosst, 2018; Hinton, 2021), seeking to overcome the inability of deep convolutional neural networks (CNNs) (Krizhevsky, Sutskever, & Hinton, 2012) which are unable to explain the images they classify in the way humans do, in terms of objects, parts and their locations.

Reference Frames Problem. In a parallel line of research, Hawkins and colleagues (Hawkins, 2021; Lewis, Purdy, Ahmad, & Hawkins, 2019) (see also (George & Hawkins, 2009; George et al., 2017)) have taken inspiration from the cortex and "grid cells" to propose that the brain uses object-centered reference frames to represent objects, spatial environments and even abstract concepts. The question of how such reference frames can be learned and used in a nested manner for hierarchical recognition and reasoning has remained open.

Integrated State-Action Hierarchy Learning Problem. A considerable literature exists on hierarchical reinforcement learning (see (Hutsebaut-Buysse, Mets, & Latré, 2022) for a recent survey), where the goal is to make traditional reinforcement learning (RL) algorithms more efficient through state and/or action abstraction. A particularly popular approach is to use options (Sutton, Precup, & Singh, 1999; Bacon, Harb, & Precup, 2016), which are abstract actions which can be selected in particular states (in the option's initiation set) and whose execution executes a sequence of primitive actions as prescribed by the option's lower level policy. The problem of simultaneously learning state and action abstraction hierarchies has remained relatively less explored.

Contributions of the Paper. The APC model addresses all

In M. Goldwater, F. K. Anggoro, B. K. Hayes, & D. C. Ong (Eds.), *Proceedings of the 45th Annual Conference of the Cognitive Science Society.*

three problems above in a unified manner using state/action embeddings and hypernetworks (Ha, Dai, & Le, 2017) to *dynamically generate and generalize* over state and action networks at multiple hierarchical levels. The APC model contributes to a number of lines of research not connected before: (1) Perception, Predictive Coding, and Reference Frame Learning: APC extends predictive coding and related neuroscience models of brain function (Rao & Ballard, 1999; Friston & Kiebel, 2009; Jiang & Rao, 2022b) to hierarchical sensory-motor inference and learning, and connects these to learning nested reference frames (Hawkins, 2021) for perception and cognition; (2) Attention Models: APC extends previous hard attention models such as the Recurrent Attention Model (RAM) (Mnih, Heess, Graves, & Kavukcuoglu, 2014) and Attend-Infer-Repeat (AIR) (Eslami et al., 2016) by learning structured hierarchical strategies for sampling the visual scene; (3) Hierarchical Planning and Reinforcement Learning: APC contributes to hierarchical planning/reinforcement learning (Hutsebaut-Buysse et al., 2022; Botvinick, Niv, & Barto, 2009) by proposing a new way of simultaneously learning abstract macro-actions or options (Sutton et al., 1999) and abstract states. Our approach brings us a step closer towards solving an important challenge in both AI and cognitive science (Lake, Ullman, Tenenbaum, & Gershman, 2017): how can neural networks learn hierarchical compositional representations that allow new concepts to be created, recognized and learned?

Active Predictive Coding

The APC model implements a hierarchical version of the traditional Partially Observable Markov Decision Process (POMDP) (Kaelbling, Littman, & Cassandra, 1998; Rao, 2010). Figure 1A shows the canonical APC generative module. The module consists of (1) a *higher level state embedding vector* $\mathbf{r}^{(i+1)}$ at level $i+1$, which uses a function H_s^i (implemented as a *hypernetwork* (Ha et al., 2017) to generate a lower level state transition *function* f_s^i (implemented as an RNN), and (2) a *higher level action embedding vector* $\mathbf{a}^{(i+1)}$, which uses a function (hypernetwork) H_a^i to generate a lower level *option/policy function* f_a^i (implemented as an RNN). In the brain, a weight-modulation version of hypernetworks (Jiang & Rao, 2022a, 2023) could be implemented via the biological mechanism of gain modulation of cortical neurons (Larkum, Senn, & Lüscher, 2004; Ferguson & Cardin, 2020). The state and action networks at the lower level are generated independently (by the higher level state/action embedding vectors) but exchange information horizontally within each level as shown in Figure 1C. In our current implementation, the lower level RNNs execute for a fixed number of time steps before returning control back to the higher level. For the present paper, we focus on a two-level model (with a top level and bottom level) as shown in Figures 1B-C.

Inference in the Active Predictive Coding Model

Inference involves estimating the state and action vectors at multiple levels based on the sequence of inputs produced by

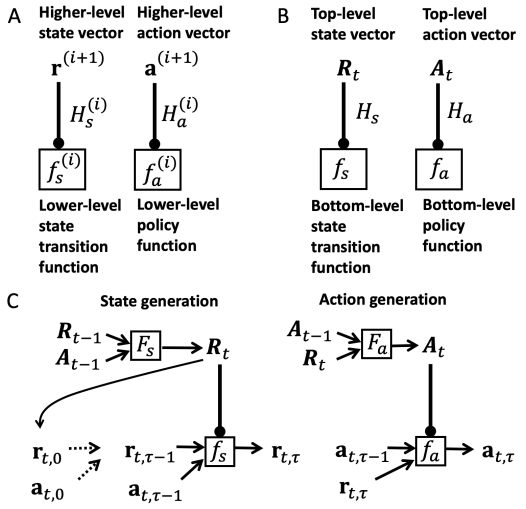


Figure 1: **Active Predictive Coding Generative Module:** (A) Canonical APC generative module. Lower level functions are generated via hypernetworks based on current higher level state and action embedding vectors. All functions (in boxes) are implemented as recurrent neural networks (RNNs). Arrows with circular terminations generate function parameters (here, neural network weights and biases). (B) Two-level model used in this paper. (C) Generation of states/actions in a 2-level model based on states/actions. Prediction errors are not shown in this generative model (see (Rao et al., 2022) for more detailed figures).

interacting with the environment in the context of a particular task or goal. The top level runs for T_2 steps (referred to as "macro-steps"). For each macro-step, the bottom level runs for T_1 "micro-steps". As shown in Figure 1C, F_s, F_a are the top level state and action networks respectively, and R_t, A_t are the recurrent activity vectors of these networks (i.e., the top level state and action embedding vectors) at macro-step t. We use the notation $f(;\theta)$ to denote a network parameterized by $\theta = \{W_l, b_l\}_{l=1}^{L}$, the weight matrices and biases for all the layers. As shown in Figure 1C, the bottom level state and action RNNs are denoted by $f_s(;\theta_s)$ and $f_a(;\theta_a)$, while their activity vectors are denoted by $r_{t,\tau}$ and $a_{t,\tau}$ respectively (t ranges over macro-steps, τ over micro-steps).

At each macro-step t, the top level state RNN F_s produces a new state embedding vector R_t based on the previous state and action embedding vectors. This higher level state R_t defines a new "reference frame" for the lower level to operate over as follows: R_t is fed as input to the state hypernetwork H_s to generate the lower level parameters $\theta_s(t) = H_s(R_t)$ specifying a dynamically generated bottom-level state RNN characterizing the state transition dynamics locally (e.g., local parts and their transformations in vision (see Application 1 below), navigation dynamics in a local region of a building (see Application 2 below), etc.). The current state R_t is also input to the action/policy RNN F_a which outputs an action embedding vector A_t (a macro-action/option/sub-goal) appropriate for the current task/goal (Figure 1C). This embedding vec-

tor is used as input to a non-linear function, implemented by the hypernetwork H_a, to dynamically generate the parameters $\theta_a(t) = H_a(A_t)$ of the lower-level action RNN, which implements a policy to generate primitive actions suitable for achieving the sub-goal associated with A_t.

At the beginning of each micro-step, the higher-level state R_t is used to initialize the bottom-level state vector via a small feedforward network Init_s to produce $r_{t,0} = \text{Init}_s(R_t)$. Each micro-step proceeds in a manner similar to a macro-step. The bottom-level action RNN produces the current action $a_{t,\tau}$ based on the current lower level state and previous action (Figure 1C lower right). This action (e.g., sensor/body movement) results in a new input being generated by the environment for the bottom (and possibly higher) state network.

To predict this new input, the lower-level state vector $r_{t,\tau}$ is fed to a generic decoder network D to generate the prediction $\hat{I}_{t,\tau}$. This predicted input is compared to the actual input to generate a prediction error $\varepsilon_{t,\tau} = I_{t,\tau} - \hat{I}_{t,\tau}$. Following the predictive coding model (Rao & Ballard, 1999), the prediction error is used to update the state vector via the state network: $r_{t,\tau+1} = f_s(r_{t,\tau}, \varepsilon_{t,\tau}, a_{t,\tau}; \theta_{(s)}(t))$.

At the end of each macro-step (after T_1 bottom-level micro-steps have finished executing), the top level state RNN activity vector is updated using the final bottom-level state vector: $R_{t+1} = F_s(R_t, A_t, \rho_s(r_{t,T_1}))$ where $\rho_s()$ is a single-layer state "feedback" network. The top-level action RNN then produces the action vector A_{t+1} based on state R_{t+1} and lower level feedback $\rho_a(a_{t,T_1})$, and the process continues.

Training the Active Predictive Coding Model

Since the state networks are task-agnostic and geared towards capturing the dynamics of the world, they are trained using self-supervised learning by minimizing prediction errors (here, via backpropagation). The action networks are trained to minimize total expected task loss: this can be done using either reinforcement learning or planning with the help of the state networks. In Application 1 below, we illustrate the use of the REINFORCE algorithm (Williams, 1992) (with backpropagation) while in Application 2, we illustrate the use of planning but the APC framework is flexible and allows either approach for estimating actions.

Application 1: Visual Perception

A long standing problem in vision and cognitive science is (Hinton, 2021): how can neural networks learn intrinsic references frames for objects and concepts, and parse inputs (e.g., images) into part-whole hierarchies? Human vision provides an important clue. Unlike convolutional nets which need to process an entire scene, human vision is an active sensory-motor process, sampling the scene via eye movements to move the high-resolution fovea to task-relevant locations, accumulating evidence for or against competing visual hypotheses (Liversedge, Gilchrist, & Everling, 2011; van Gompel, Fischer, Murray, & Hill, 2007). The APC model is well-suited to emulating the sensory-motor nature of human vision, given its integrated state and action networks.

For visual perception and part-whole learning, the actions in the APC model emulate eye movements (or "attention") by moving a "glimpse sensor" (Mnih et al., 2014) which extracts high-resolution information about a small part of a larger input image. Given an input image I (of size $N \times N$ pixels), this sensor, G, takes in a location l and a fixed scale fraction m, and extracts a square glimpse/patch $g = G(I, l, m)$ centered at l and of size $(mN) \times (mN)$. At each macro-step t, the top-level action vector A_t generates two values: (a) a location L_t and (b) a macro-action (or option) z_t. The location L_t is used to restrict the bottom level to a sub-region $I_t^{(1)} = G(I, L_t, M)$ corresponding to a new frame of reference selected by the top level within the input image. The option z_t, which operates over this frame of reference, is used as an embedding vector input to the hypernetwork H_a to generate the parameters of the bottom-level action RNN. For exploration during reinforcement learning, we treat the output of the location network as a mean value $\bar{L}_t$ and add Gaussian noise with fixed variance to sample an actual location: $L_t = \bar{L}_t + \eta$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. We do the same for the option z_t. Based on the current bottom-level state and action, the bottom-level action RNN generates a new action $a_{t,\tau}$. A location $l_{t,\tau}$ is chosen as a function of $a_{t,\tau}$, resulting in a new glimpse image $g_{t,\tau} = G(I_t^{(1)}, l_{t,\tau}, m)$ of scale m centered around $l_{t,\tau}$ and yielding a nested reference frame within the larger reference frame of $I_t^{(1)}$ specified by the higher level. The bottom level follows the same Gaussian noise-based exploration strategy for sampling locations as the top level. The bottom-level state vector $r_{t,\tau}$, along with locations L_t and $l_{t,\tau}$, are fed to a generic decoder network D to generate the predicted glimpse $\hat{g}_{t,\tau}$. Following the predictive coding model (Rao & Ballard, 1999), the resulting prediction error $\varepsilon_{t,\tau} = g_{t,\tau} - \hat{g}_{t,\tau}$ is used to update the state vector: $r_{t,\tau+1} = f_s(r_{t,\tau}, \varepsilon_{t,\tau}, l_t; \theta_{(s)}(t))$. For the results below, the state networks at both levels were trained to minimize image prediction errors while the action networks were trained using reinforcement learning (the REINFORCE algorithm (Williams, 1992); reward based on image reconstruction error - see (Rao et al., 2022)).

Results

We first tested the APC model on the task of sequential part/location prediction and image reconstruction of objects in the following datasets: (a) MNIST: Original MNIST dataset of 10 classes of handwritten digits. (b) Fashion-MNIST: Instead of digits, the dataset consists of 10 classes of clothing items. (c) Omniglot: 1623 hand-written characters from 50 alphabets, with 20 samples per character. For our APC models, we used 3 macro- and 3 micro-steps (except 4 macro-steps for Omniglot). A single dense layer, together with an initial random glimpse was used to initialize the state and action vectors of the top level.

Parsing Images and Perceptual Stability. Figure 2 shows an example of a learned parsing strategy by a two-level APC model for an MNIST digit. The top-level learned to cover the input image sufficiently while the bottom level learned to

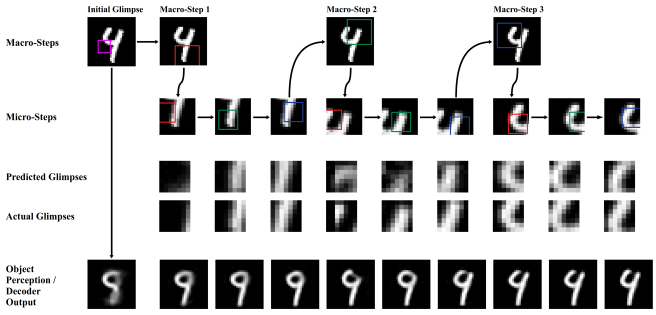


Figure 2: **Learned Two-Level Parsing Strategy and an Illustration of Perceptual Stability in the presence of Eye Movements:** 1st row: Initialization glimpse (purple box) and sampled top-level reference frames (red, green, blue boxes), 2nd row: Sampled bottom level parts within each top-level frame, 3rd & 4th rows: Predicted versus actual parts/glimpses, and 5th row: "Perception" of the model (object reconstructed from current network state) over time.

parse "sub-parts" inside the reference frame computed by the higher level. Figure 2 provides a proof-of-concept demonstration of how perception can appear stable despite dramatic changes in retinal images as the eyes move to sample a scene: the last row shows how the model maintains a visual hypothesis that is gradually refined and does not exhibit the kind of rapid changes seen in the sampled images ("Actual Glimpses" in Figure 2). Figure 3 shows a learned part-whole hierarchy for an MNIST input, in the form of a parse tree of "parts" and "sub-parts" (strokes and mini-strokes) with locations. The model learns different parsing strategies for different classes of objects (Figure 4). Comparison to human image parsing remains a direction for future research.

Prediction of Parts and Pattern Completion. To investigate the predictive and generative ability of the model, we had the model "hallucinate" different parts of an object by setting the prediction error input to the lower level network to zero. This disconnects the model from the input, forcing it to predict the next sequence of parts and "complete" the object. Figure 5a shows that the model has learned to generate plausible predictions of parts given an initial glimpse.

Transfer Learning. We tested transfer learning for reconstruction of unseen character classes for the Omniglot dataset. We trained a two-level APC model to reconstruct examples from 85% of classes from each Omniglot alphabets. The held-out classes were used to test transfer: the trained model had to generate new programs (via the state and action hypernets) to predict parts for new character classes for each alphabet. The model successfully performed this task (Table 1, Figure 5b).

Ablation Studies. To test the utility of having 2 levels of abstraction, we compared (on held-out samples) the reconstruction performance of the 2-level APC model (APC-2) to a 1-level model (APC-1) and a Randomized Baseline model (RB), which samples glimpses (same size as APC-1 and APC-2) from T i.i.d. locations (T same as for APC-1 and

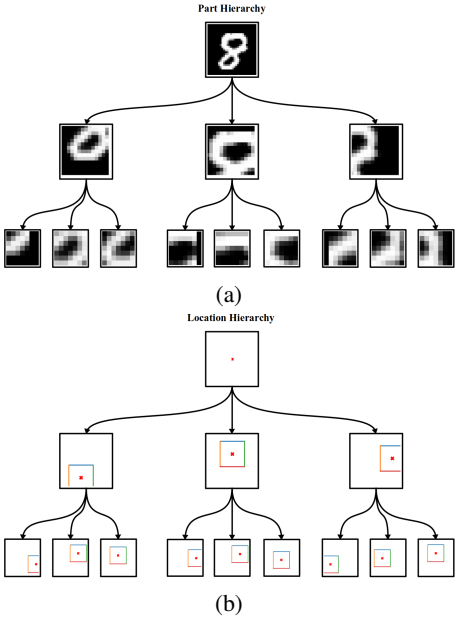


Figure 3: **Example Parse Tree with Inferred Locations of Parts:** Hierarchy of (a) sampled parts and (b) sampled locations, inducing a hierarchy of reference frames.

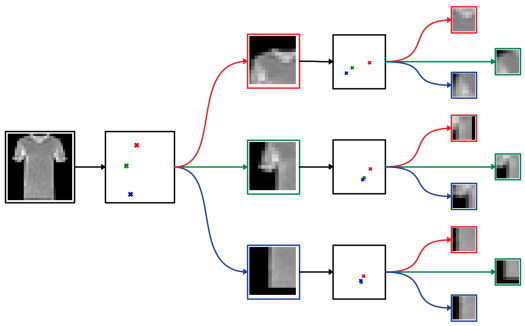
	MNIST	FMNIST	Om-Tst	Om-Trn
RB	0.0120	0.0145	0.0307	0.0301
APC-1	0.0114	**0.0138**	0.0324	0.0323
APC-2	**0.0085**	**0.0138**	**0.0227**	**0.0226**

Table 1: **Ablation Studies: Reconstruction Mean-Squared-Error (per pixel) for Different Models Across Datasets:** See text for details. FMNIST, Om-Tst and Om-Trn denote Fashion-MNIST, the Omniglot test and transfer datasets respectively.

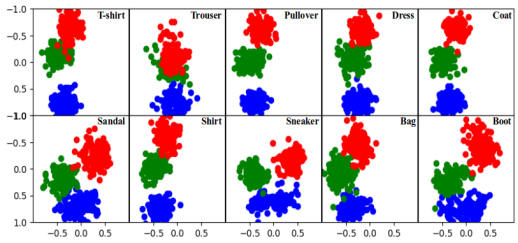
APC-2, i.e., 9 for MNIST/FMNIST, 12 for Omniglot), extracts an average feature vector and feeds this to a feedforward network to reconstruct the image. As shown in Table 1, APC-2 outperforms or matches APC-1 and RB. APC-2 also outperforms RB and APC-1 on the Ommiglot transfer learning task ("Om-Trn" in Table 1).

Application 2: Hierarchical Planning

We now show that the same APC framework used for learning part-whole hierarchies can also be used for learning hierarchical world models for efficient planning. We introduce a new compositional, scalable "multi-rooms" building navigation task to illustrate this. Consider the problem of navigating from any starting location to any goal location in a large building environment such as the one in Figure 6A (gray: walls, blue circle: agent, green square: current goal). In the traditional non-hierarchical RL approach, the states are the discrete locations in the grid, and actions are going north (N), east (E), south (S) or west (W). A large reward (+10) is received at the goal location, with a small negative reward

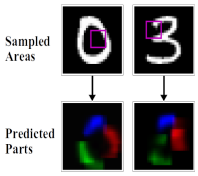


(a)

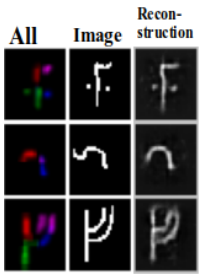


(b)

Figure 4: **Class-Based Hierarchical Representation of Object Parts and Locations:** (a) Parts and sub-parts recognized by a 2-level APC network trained on the Fashion- MNIST dataset for an input image of a T-shirt. The order of sampled locations within each frame of reference is 1st: red, 2nd: green and 3rd: blue. (b) Each panel shows the top-level part locations selected by the trained APC network in (a) for all classes. Note the differences in the network's action strategies between vertically symmetric items and footwear.



(a)



(b)

Figure 5: **Prediction of Parts, Pattern Completion and Transfer Learning:** (a) Given an initialization glimpse (purple boxes on the "0" and "3" images), an APC model trained on MNIST predicts its best guess of the parts of the object and their locations (colored segments in row below). (b) APC model trained on Omniglot can transfer its learned knowledge to predict parts of previously unseen character classes. First column: all the predicted parts. Middle column: input from a novel class. Last column: APC model reconstruction.

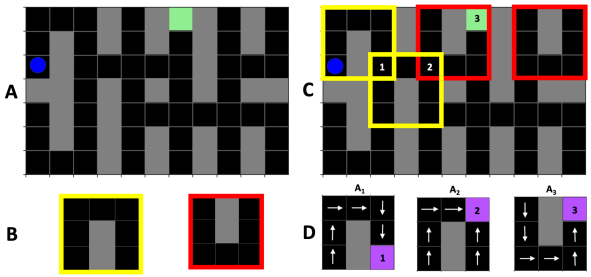


Figure 6: **The Multi-Rooms Building Navigation Problem and State-Action Hierarchy**. The problem of navigating in a large building (A) (blue: agent location, gray: walls, green: goal) can be reduced to planning using high-level states ((B) and (C)) and high-level actions (D). See text for details.

(-0.1) for each action to encourage shortest paths.

Problems with traditional RL. (1) Sample inefficiency: As the environment gets larger, the number of interactions with the environment required to learn the value function explodes, (2) Risk of catastrophic consequences: Taking actual actions in the real-world to estimate the value function might have catastrophic consequences (injury or death), and (3) Inflexibility: A new goal requires learning a new value function. *How the APC model solves these problems.* First, note that just as an object (e.g., an MNIST digit) consists of the same parts (e.g., strokes) occurring at different locations, the multi-rooms environment in Figure 6A is also made up of the same two components ("Room types" R1 and R2), shown in Figure 6B, occurring at different locations (some example locations highlighted by yellow and red boxes in Figure 6C). These components form part of the higher-level states in the APC and are defined by state embedding vectors R1 and R2, which can be trained to generate, via the hypernet H_s (Figure 1B), the lower-level transition functions f_s for rooms R1 and R2 respectively. Next, similar to how the APC model was able to reconstruct an image using top-level action embedding vectors to generate policies and actions (locations) to compose parts using strokes, the APC model can compute top-level action embedding vectors $\mathbf{A}_i$ (option vectors) for the multi-rooms world that generate, via hypernet H_a (Figure 1B), bottom-level policies f_a which produce primitive actions (N, E, S, W) to reach a goal i encoded by $\mathbf{A}_i$.

Local reference frames allow policy re-use and transfer. Figure 6D illustrates the bottom-level policies for three such action embedding vectors $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$, which generate policies for reaching goal locations 1, 2, and 3 respectively. Note that the $\mathbf{A}_i$ are defined with respect to higher-level state R1 or R2. Defining these policies to operate within the local reference frame of the higher-level state R1 or R2 (regardless of global location in the building) confers the APC model with enormous flexibility because the same policy can be re-used at multiple locations to solve local tasks (here, reach sub-goals within R1 or R2). For example, to solve the navigation problem in Figure 6C, the APC model only needs to plan and execute 3 higher-level actions or options: $\mathbf{A}_1$ followed by $\mathbf{A}_2$ followed by $\mathbf{A}_3$, compared to planning a sequence of 12 lower-level actions to reach the same goal. Finally, since

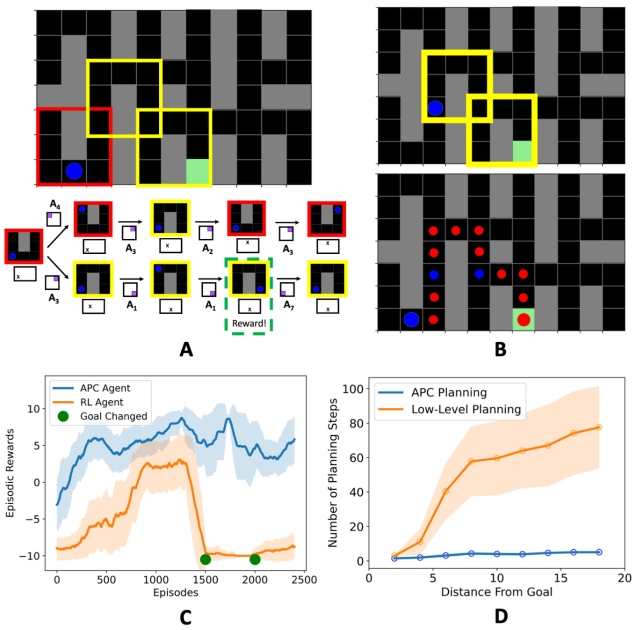


Figure 7: **Planning and Model Predictive Control**. To navigate to the green goal in (A, top panel), the agent (blue) uses its learned high-level state network to sample N high-level state-action trajectories ($N = 2$ in (A, bottom panel)), picks the sequence with highest total reward, executes this sequence's first action to reach the location in (B, top panel), and repeats to reach the goal with 3 high-level actions (B, bottom panel). Small red dot: intermediate location; small blue dot: intermediate goal. In (A, bottom panel), high-level state is depicted by predicted R1 or R2 image and its location X in the global frame, action by its goal (purple) in a square local frame. (C), (D) APC model performance: see text for details.

the $\mathbf{A}_i$ embedding space of options is continuous, there is an unprecedented opportunity for the APC model to exploit properties of this space (such as smoothness) to interpolate or extrapolate to create and explore new options for transfer learning; this possibility will be explored in future work.

Results

For simplicity, we assume the higher level states capture 3×3 local reference frames and are defined by an embedding vector generating the transition function for "room type" R1 or R2 , along with the location for this local reference frame in the global frame of the building. The lower-level action network is trained to map a higher-level action embedding vector $\mathbf{A}_i$ to a lower-level policy that navigates the agent to a particular goal location i within R1 or R2. For the current example, eight embedding vectors $A_1, \ldots, A_8$ were trained, using REINFORCE-based RL (Williams, 1992), to generate via the hypernet H_a eight lower-level policies to navigate to each of the four corners of room types R1 and R2. The higher-level state network F_s was trained to predict the next higher-level state (decoded as an image of room type R1 or R2, plus its location) given the current higher-level state and higher-level action. The trained higher-level state network F_s was used for planning at each step a sequence of 4 higher-level actions using random-sampling shooting model-predictive control (MPC) (Richards, 2004): random state-action trajectories of length 4 were generated using F_s by starting from the current state and picking one of the four random actions A_i for each next state; the action sequence with the highest total reward was selected and its first action was executed (Figures 7A and B).

We compared the two-level APC model with both a heuristic lower-level-only planning algorithm and a REINFORCE-based RL algorithm using primitive states and actions. The task involved navigating to a randomly selected goal location in a building environment (as in Figure 6A), with the goal location changing after some number of episodes. Figure 7C shows how the APC model, after an initial period spent on learning the hypernet H_a to generate the lower-level options, is able to cope with goal changes and successfully navigate to each new goal by sequencing high-level actions (+10 reward for goal; -0.1 per primitive action). The RL algorithm, on the other hand, experiences a drop in performance after a goal change and does not recover even after 500 episodes. Figure 7D demonstrates the efficacy of APC's higher-level planning compared to lower-level planning (MPC using random sequences of 4 primitive future actions; Euclidean distance heuristic): the average number of planning steps to reach the goal increases dramatically for larger distances from the goal for lower-level compared to higher-level planning.

Conclusion

Our results represent a first proof-of-concept demonstration of a unified approach to the problems of part-whole learning (Hinton, 2021), learning nested reference frames (Hawkins, 2021), and integrated state-action hierarchy learning for compositional problem solving (Hutsebaut-Buysse et al., 2022). Our APC model is inspired by the growing evidence for predictive sensory-motor processing in the cortex (Schneider, Sundararajan, & Mooney, 2018; Keller & Mrsic-Flogel, 2018). The APC model for vision employs "eye movements" for visual sampling, and performs end-to-end learning and parsing of part-whole hierarchies from images. We also showed how the same APC framework provides a flexible approach to hierarchical planning and action selection.

Our results relied on assumptions such as a fixed number of time steps at each level, a two-level hierarchy, hard-coded glimpse operators, and pre-identified higher-level states and actions for planning. Future work will involve relaxing these assumptions, comparing to other part-whole learning approaches with different metrics, and employing more sophisticated planning and RL methods for scaling to more complex tasks. Given the growing interest in predictive coding as a model for cortical computation, the diverse applicability of active predictive coding hinted at by our results support the intriguing hypothesis of a common computational principle operating across the cortex (Mountcastle, 1978; Rao, 2022).

Acknowledgements

This material is based upon work supported by National Science Foundation (NSF) EFRI grant no. 2223495, DARPA grant no. HR001120C0021, a Weill Neurohub Investigator grant, and a "Frameworks" grant from the Templeton World Charity Foundation. The opinions expressed in this publication are those of the authors and do not necessarily reflect the views of the funders.

References

Bacon, P.-L., Harb, J., & Precup, D. (2016). The option-critic architecture. *CoRR*, *abs/1609.05140*.

Botvinick, M. M., Niv, Y., & Barto, A. G. (2009). Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *Cognition*, *113*(3), 262-280.

Eslami, S. M. A., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Kavukcuoglu, K., & Hinton, G. E. (2016). Attend, infer, repeat: Fast scene understanding with generative models. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 29). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2016/file/52947e0ade57a09e4a1386d08f17b656-Paper.pdf

Ferguson, K. A., & Cardin, J. A. (2020). Mechanisms underlying gain modulation in the cortex. *Nature Reviews Neuroscience*, *21*(2), 80–92. Retrieved 2022-12-01, from https://www.nature.com/articles/s41583-019-0253-y doi: 10.1038/s41583-019-0253-y

Friston, K., & Kiebel, S. (2009). Predictive coding under the free-energy principle. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, *364*, 1211-21. doi: 10.1098/rstb.2008.0300

George, D., & Hawkins, J. (2009). Towards a mathematical theory of cortical micro-circuits. *PLoS Computational Biology*, *5*, e1000532. doi: 10.1371/journal.pcbi.1000532

George, D., Lehrach, W., Kansky, K., Lázaro-Gredilla, M., Laan, C., Marthi, B., ... Phoenix, D. S. (2017). A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. *Science*, *358*(6368), eaag2612. Retrieved from https://www.science.org/doi/abs/10.1126/science.aag2612

Ha, D., Dai, A. M., & Le, Q. V. (2017). Hypernetworks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net. Retrieved from https://openreview.net/forum?id=rkpACe1lx

Hawkins, J. (2021). *A thousand brains: A new theory of intelligence*. Basic Books. Retrieved from https://books.google.com/books?id=hYrvDwAAQBAJ

Hinton, G. E. (2021). How to represent part-whole hierarchies in a neural network. *CoRR*, *abs/2102.12627*. Retrieved from https://arxiv.org/abs/2102.12627

Hinton, G. E., Sabour, S., & Frosst, N. (2018). Matrix capsules with EM routing. In *6th International Conference on Learning Representations, ICLR 2018*. OpenReview.net. Retrieved from https://openreview.net/forum?id=HJWLfGWRb

Hutsebaut-Buysse, M., Mets, K., & Latré, S. (2022). Hierarchical reinforcement learning: A survey and open research challenges. *Mach. Learn. Knowl. Extr*, *100*, 172–221.

Jiang, L. P., & Rao, R. P. N. (2022a). Dynamic predictive coding: A new model of hierarchical sequence learning and prediction in the cortex. *bioRxiv 2022.06.23.497415v3*. Retrieved from https://www.biorxiv.org/content/10.1101/2022.06.23.497415v3

Jiang, L. P., & Rao, R. P. N. (2022b). Predictive coding theories of cortical function. *Oxford Research Encyclopedia of Neuroscience*.

Jiang, L. P., & Rao, R. P. N. (2023). Dynamic predictive coding explains both prediction and postdiction in visual motion perception. *Proceedings of the 2023 Annual Meeting of the Cognitive Science Society (CogSci 2023)*.

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artif. Intell.*, *101*(1-2), 99–134. Retrieved from https://doi.org/10.1016/S0004-3702(98)00023-X

Keller, G. B., & Mrsic-Flogel, T. D. (2018). Predictive processing: A canonical cortical computation. *Neuron*, *100*(2), 424-435. Retrieved from https://www.sciencedirect.com/science/article/pii/S0896627318308572 doi: https://doi.org/10.1016/j.neuron.2018.10.003

Kosiorek, A., Sabour, S., Teh, Y. W., & Hinton, G. E. (2019). Stacked capsule autoencoders. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2019/file/2e0d41e02c5be4668ec1b0730b3346a8-Paper.pdf

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 25). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*, e253. doi: 10.1017/S0140525X16001837

Larkum, M. E., Senn, W., & Lüscher, H.-R. (2004). Top-down Dendritic Input Increases the Gain of Layer 5 Pyramidal Neurons. *Cerebral Cortex*, *14*(10), 1059–1070. Retrieved 2022-12-01, from https://doi.org/10.1093/cercor/bhh065

Lewis, M., Purdy, S., Ahmad, S., & Hawkins, J. (2019). Locations in the neocortex: A theory of sensorimotor object recognition using cortical grid cells. *Frontiers in Neural*

Circuits, *13*. Retrieved from https://www.frontiersin.org/article/10.3389/fncir.2019.00022

Liversedge, S., Gilchrist, I., & Everling, S. (Eds.). (2011). *The Oxford Handbook of Eye Movements*. Oxford University Press.

Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2014/file/09c6c3783b4a70054da74f2538ed47c6-Paper.pdf

Mountcastle, V. (1978). An organizing principle for cerebral function: the unit model and the distributed system. In G. Edelman & V. Mountcastle (Eds.), *The Mindful Brain* (p. 7–50). Cambridge, MA: MIT Press.

Rao, R. P. N. (2010). Decision making under uncertainty: a neural model based on partially observable markov decision processes. *Frontiers in computational neuroscience*, *4*, 146.

Rao, R. P. N. (2022). A sensory-motor theory of the neocortex based on active predictive coding. *bioRxiv 2022.12.30.522267*.

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

Rao, R. P. N., Gklezakos, D. C., & Sathish, V. (2022). Active predictive coding: A unified neural framework for learning hierarchical world models for perception and planning. *arXiv:2210.13461*.

Richards, A. (2004). *Robust constrained model predictive control*. Unpublished doctoral dissertation, MIT.

Sabour, S., Frosst, N., & Hinton, G. E. (2017). *Dynamic routing between capsules.* Retrieved from http://arxiv.org/abs/1710.09829 (cite arxiv:1710.09829)

Schneider, D. M., Sundararajan, J., & Mooney, R. (2018). A cortical filter that learns to suppress the acoustic consequences of movement. *Nature*, *561*(7723), 391-395. Retrieved from https://doi.org/10.1038/s41586-018-0520-5

Sherman, S. M., & Guillery, R. W. (2013). *Functional connections of cortical areas: A new view from the thalamus*. MIT Press.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*(1), 181-211. Retrieved from https://www.sciencedirect.com/science/article/pii/S0004370299000521

van Gompel, R. P. G., Fischer, M. H., Murray, W. S., & Hill, R. L. (Eds.). (2007). *Eye movements: A window on mind and brain*. Elsevier.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*, 229-256.