Combined Model for Partially-Observable and Non-Observable Task Switching: Solving Hierarchical Reinforcement Learning Problems Statically and Dynamically with Transfer Learning

Nibraas Khan (nak2z@mtmail.mtsu.edu) and Joshua Phillips (Joshua.Phillips@mtsu.edu)

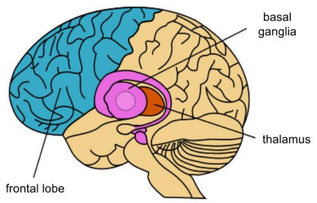
Department of Computer Science, Middle Tennessee State University

Abstract

An integral function of fully autonomous robots and humans is the ability to focus attention on a few relevant percepts to reach a certain goal while disregarding irrelevant percepts. Humans and animals rely on the interactions between the PreFrontal Cortex (PFC) and the Basal Ganglia (BG) to achieve this focus called Working Memory (WM). The Working Memory Toolkit (WMtk) was developed based on a computational neuroscience model of this phenomenon with Temporal Difference (TD) Learning for autonomous systems. Recent adaptations of the toolkit either utilize Abstract Task Representations (ATRs) to solve Non-Observable (NO) tasks or storage of past input features to solve Partially-Observable (PO) tasks, but not both. We propose a new model, PONOWMtk, which combines both approaches, ATRs and input storage, with a static or dynamic number of ATRs. The results of our experiments show that PONOWMtk performs effectively for tasks that exhibit PO, NO, or both properties.

Introduction

Robots, specifically autonomous robots, needs to be able to see the world around them to be able to understand and behave like humanoids. We as humans have our own way of perceiving the world around us – through the interactions between out Basal Ganglia and the Pre-Frontal Cortex.



For autonomous systems to be able to solve the kind of complex tasks that we expect from a true General Artificial Intelligent System, they need to understand to world around them. So, we propose a toolkit that will allow robots the ability to *think*. With thought, our model allows a robot to solve complex problems regardless of if the problem presents external stimuli for for the entire task like image recognition (fully observable), presents external stimuli only for a certain amount like self-driving car (partially observable), or no stimuli at all like the Wisconsin Card Sorting task (non observable).

Our model is able to replicate the interactions using Temporal Difference Learning, Neural Networks, Transfer Learning, Holographic reduced representations, and Abstract Task Representations.

Background

Our model will take two existing toolkits and combine them (with additional features added on top).

The first constituent of our model is the Working Memory Toolkit:

- Solves Fully and Partially observable tasks
- Creates a memory for the each everything it perceives
- As it solves the task, it forms a value for each memory corresponding to the goal
- Stores the values in a neural network

The second constituent is the N-task learning toolkit:

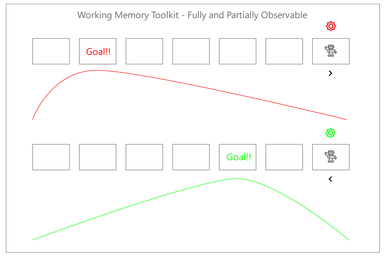
- Solves Non observable tasks
- Looks at the environment through lenses or ATRs which correspond to the tasks
    - i.e. When making coffee, it looks at the world as though the only thing it can do is make coffee
- As it solves the tasks, it associates each lens with a different task
- It can change lenses based on whether the current lens leads to reward (completion of the task)

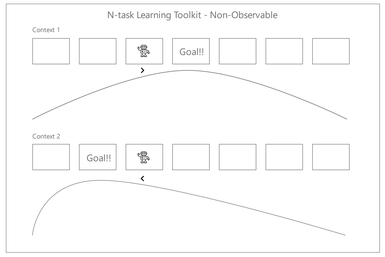
The two models mentioned above have significant limitations when solving complex real world tasks:

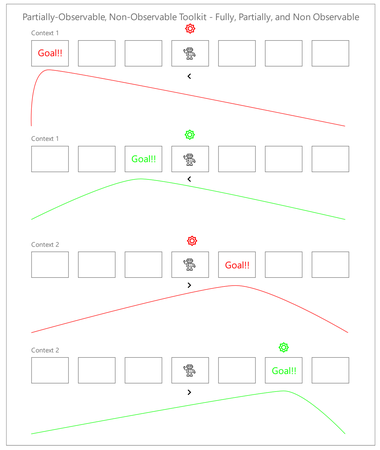
- Many real tasks have layers of Observable and Non observable features that neither model could solve
- The N-task toolkit forgets everything when it things there is another task
- Additional lens switching mechanisms are needed

Methods

For our Partially-Oberservable Non-Observable Working Memory Toolkit, we took the main components of the Working Memory Toolkit and the N-task Learning toolkit and combined them. On top of the combination, we implemented several feature for smoother learning and more success. To understand the techniques, we consider a maze task where a robot is dropped into a maze and has to find the goal state by moving left and right.







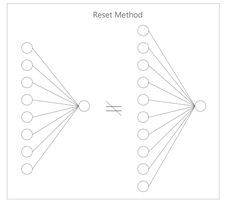
The robot icon specifies the start location, the "Goal!!" specifies the goal for that task, the color of the light specifies the external stimuli, the context label specifies the different tasks without external stimuli, the arrows specify the direction the robot will step into, and the lines specify the internal values of each state of the maze.

Working Memory Toolkit (Memory switched by stimuli)

- The robot is dropped into a random location
- A color is flashed for one timestep
- Using the color, the robot needs to use the corresponding memory to find the goal

N-task Learning Toolkit (ATRs switched by internal error)

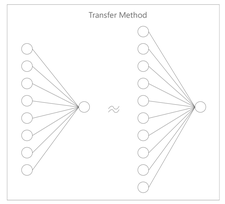
- The robot is dropped into a random location
- Without any stimuli, the robot has to find the goal
- Has to understand which context it is in by the reward it thinks it will receive at a step. Wrong memory will result in unexpected high or low rewards



Reset and Transfer Method

The robot may be presented with any number of tasks, so it needs to grow in size to be able to handle them all. There are two ways to expand the capabilities of the agent: scraping everything is has learned (Reset) or saving what it has learned (Transfer). For transfer, the robot gets all the values of the smaller network and moves them to the larger using:

$$\vec{w} \leftarrow \left[\underset{\vec{u}_{new} \in U_{new}}{\vec{u}_{new}}\right]^{-1} * \left[\underset{\vec{u}_{old} \in U_{old}}{\mathbf{v}}(\vec{u}_{old})\right]$$



Partially-Observable, Non-Observable Toolkit (Memory and ATRs switched with stimuli and internal error)

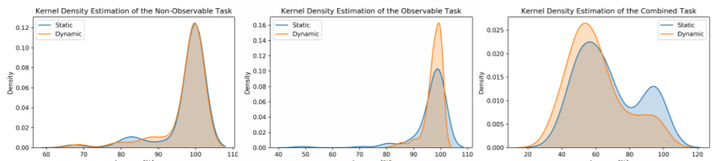
- The robot is dropped into a random location and the robot forms an internal representation (state, signal, memory, ATR, and reward) using:
  $\vec{u} = (\vec{s} \wedge \vec{p} \wedge w\vec{m} \wedge a\vec{t}r \wedge \vec{r})$
- The value of each state is calculated using a neural network using:
  $v(\vec{u}) = (\vec{u} \cdot \vec{w}) + b$
- As the robot moves, it learned using the neural network. For this, it needs to understand its errors using:
  $\delta_t = (r_t - \gamma \cdot \mathrm{v}(\vec{u}_{t+1})) - \mathrm{v}(\vec{u}_t)$
- It also needs to took at all the moves it has taken so far using:
  $\vec{e}_t = \lambda \cdot \vec{e}_{t-1} + \vec{u}_t$
- And finally it can update its weights using:
  $\vec{w}_t = \vec{w}_{t-1} + \alpha \cdot \mathrm{logmod}(\delta_t) \cdot \vec{e}_t$
- To choose a move and memory, the robot using:
  $m, c = \underset{\vec{s} \in S,\ \vec{w}\vec{m} \in WM}{argmax} (\mathrm{v}(\vec{s} \wedge \vec{p} \wedge w\vec{m} \wedge a\vec{t}r \wedge \vec{r}))$
- And chooses the ATR using:
  $atr \leftarrow \underset{a\vec{t}r \in ATR}{argmax} (\vec{s} \wedge \vec{p} \wedge w\vec{m} \wedge a\vec{t}r \wedge \vec{r})$
- The model also keeps track of a threshold that is checked when a large positive of negative error is seen in order the consider whether a new ATR is needed. The dynamic threshold if kept using:
  $t \mathrel{+}= \alpha_t \cdot \mathrm{logmod}(\delta)$

Results

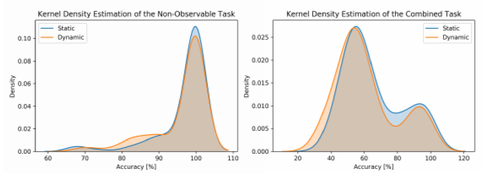
The results on the right analyses all three kinds of tasks (Fully, Partially, and Non-Observable), both the reset and transfer method, and both static and dynamic thresholds.

For both the Partially and Non-Observable tasks the model is able to solve the tasks with almost 100% accuracy due to the fact that it is leaning on its constituents. However, when the model is faced with the combined tasks, it does very bad. The kernel density estimation puts the accuracy around 50%. Our model was made for the purpose of solving these kinds of problems, but it appears that it cannot. That is because the model has not been given the chance to tune its hyper-parameters.

A combined task is extremely complex, so the model needs to tune its hyper-parameters as well as solve the problem as mentioned above. With the tuning, the model is able to achieve 90-100% accuracy, thus solving the combined task problem.



(a) NO task with reset method (b) PO task with reset method (c) PONO task with reset method



(d) NO task with transfer method (e) PONO task with transfer method

Implementation Details

There are several points that we have not gone into detail:

- Bayesian Optimization: Improving results through fine tuning the hyper-parameters of the model. The fine tuning is done
- ATR Switching: There are three main methods of switching ATRs for the model. It can switch based on positive error, negative error, or both. Each of theses methods have their benefits, but for the best results, a combination of the errors needs to be used.
- Holographic Reduced Representations: For encoding all of the information presented to the robot, it uses HRRs. HRRs are just a list of numbers drawn from a normal distribution, and each HRR is almost orthogonal to each other. With this method of encoding, we can use a simple one-layer neural network to hold all the values for all the states that the robot has encountered.

Conclusion and Future Work

From the results, it is clear that the model is able to solve the combined task. The model is able to rely on its constituents as well as take advantage of its additional features to solve all three tasks. But there are some issues that need to be addressed.

For one, the model is needs to tune its parameters as well as its weights to achieve the kind of accuracy that we expect from a model such as this. This is not a unreasonable requirement; however, it causes the training time of the model to increase dramatically. For use in robotics, the hardware and time is often limited, so the model needs to be optimized to fun on smaller systems.

The model also works using a simple one-layer neural network, so there is room for more complexity in the model. With the use of more complex model with more hidden layers, the model will able to expand its capabilities.

References

Baddeley, A. (1992). Working memory. Science, 255(5044), 556–559.

Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599.

Collins, A. G., & Frank, M. J. (2012). How much of reinforcement learning is working memory, not reinforcement learning? a behavioral, computational, and neurogenetic analysis. European Journal of Neuroscience, 35(7), 1024– 1035.

DuBois, G. M., & Phillips, J. L. (2017). Working memory concept encoding using holographic reduced representations. In Maics (pp. 137–144).

Fukuda, T., Michelini, R., Potkonjak, V., Tzafestas, S., Valavanis, K., & Vukobratovic, M. (2001, March). How far away is "artificial man". IEEE Robotics Automation Magazine, 8(1), 66-73. doi: 10.1109/100.924367

Jovanovich, M., & Phillips, J. (2018). n-task learning: Solving multiple or unknown numbers of reinforcement learning problems. In Cogsci (pp. 584–589).

Kunz, F. (2000). An introduction to temporal difference learning. In Seminar on autonomous learning systems.

O'Reilly, R. C., Noelle, D. C., Braver, T. S., & Cohen, J. D. (2002). Prefrontal cortex and dynamic categorization tasks: representational organization and neuromodulatory control. Cerebral cortex, 12(3), 246–257.

Phillips, J. L., & Noelle, D. C. (2005). A biologically inspired working memory framework for robots. In Roman 2005. ieee international workshop on robot and human interactive communication, 2005. (pp. 599–604).

Plate, T. A. (1995). Holographic reduced representations. IEEE Transactions on Neural networks, 6(3), 623–641.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Tange, O. (2011, Feb). Gnu parallel - the command-line power tool. ;login: The USENIX Magazine, 36(1), 42-47. Retrieved from http://www.gnu.org/s/parallel doi: http://dx.doi.org/10.5281/zenodo.16303

Tugcu, M., Wang, X., Hunter, J. E., Phillips, J., Noelle, D., & Wilkes, D. M. (2007). A computational neuroscience model of working memory with application to robot perceptual learning. In Third Third iasted international conference on computational intelligence (ci).

Acknowledgement

We would like to thank the Undergraduate Research Experience and Creative Activity at Middle Tennessee State University for funding this research.