State Space Decomposition and Subgoal Creation for Transfer in Deep Reinforcement Learning

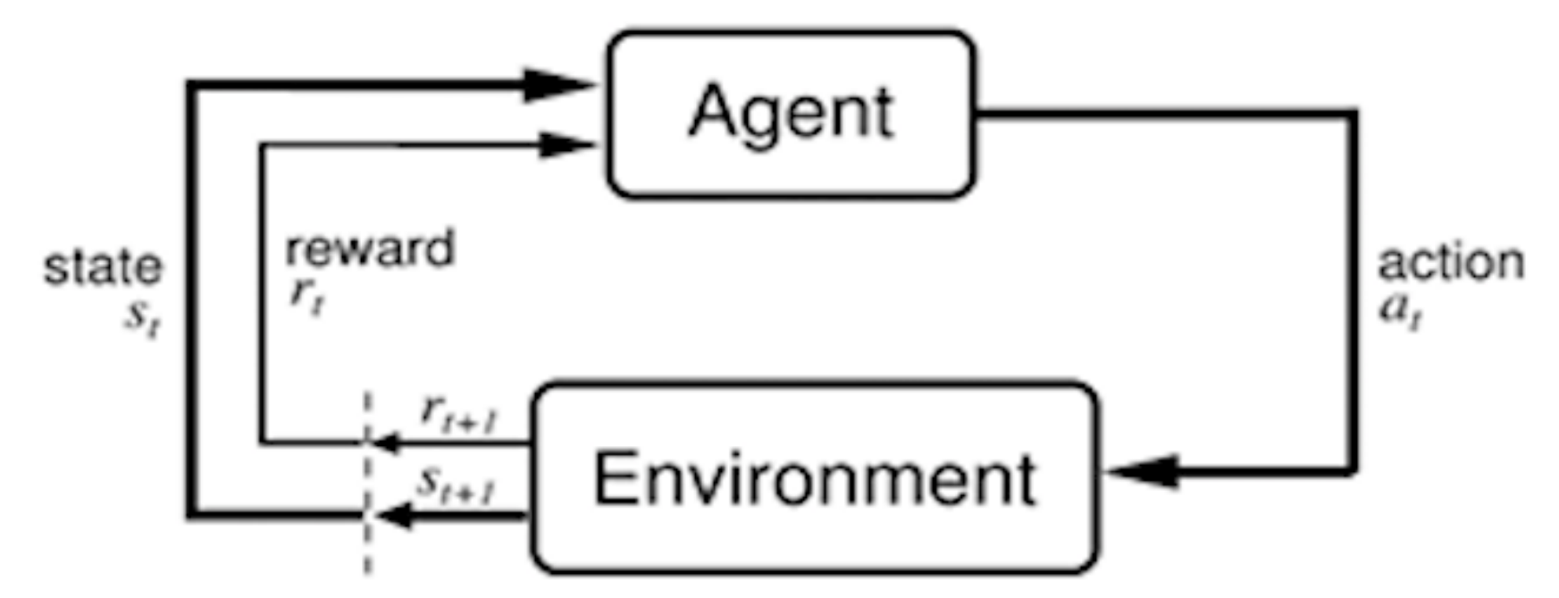
Saurabh Kumar, Himanshu Sahni, Farhan Tejani, Yannick Schroeker, Charles Isbell
College of Computing, Georgia Institute of Technology

Abstract

Typical reinforcement learning (RL) agents learn to complete tasks specified by reward functions tailored to their domain. As such, the policies they learn do not generalize even to similar domains. To address this issue, we develop a framework through which a deep RL agent learns to generalize policies from smaller, simpler domains to more complex ones using a recurrent attention mechanism. The task is presented to the agent as an image and an instruction specifying the goal. This meta-controller guides the agent towards its goal by designing a sequence of smaller sub-tasks on the part of the state space within the attention, effectively decomposing it. As a baseline, we consider a setup without attention as well. Our experiments show that the meta-controller learns to create sub-goals within the attention.

Background

Reinforcement Learning



Policy Gradient Method

A Policy: $\pi: S \rightarrow A$ maps states to a particular action the agent should take. The goal of the agent is to find a policy that maximizes the expected sum of rewards the agent receives.[1]

The REINFORCE Policy Gradient algorithm[2] updates the policy by:
Sampling trajectories
1) Increasing the log probability of "good actions"
2) Decreasing the log probability of "bad actions"

Environments

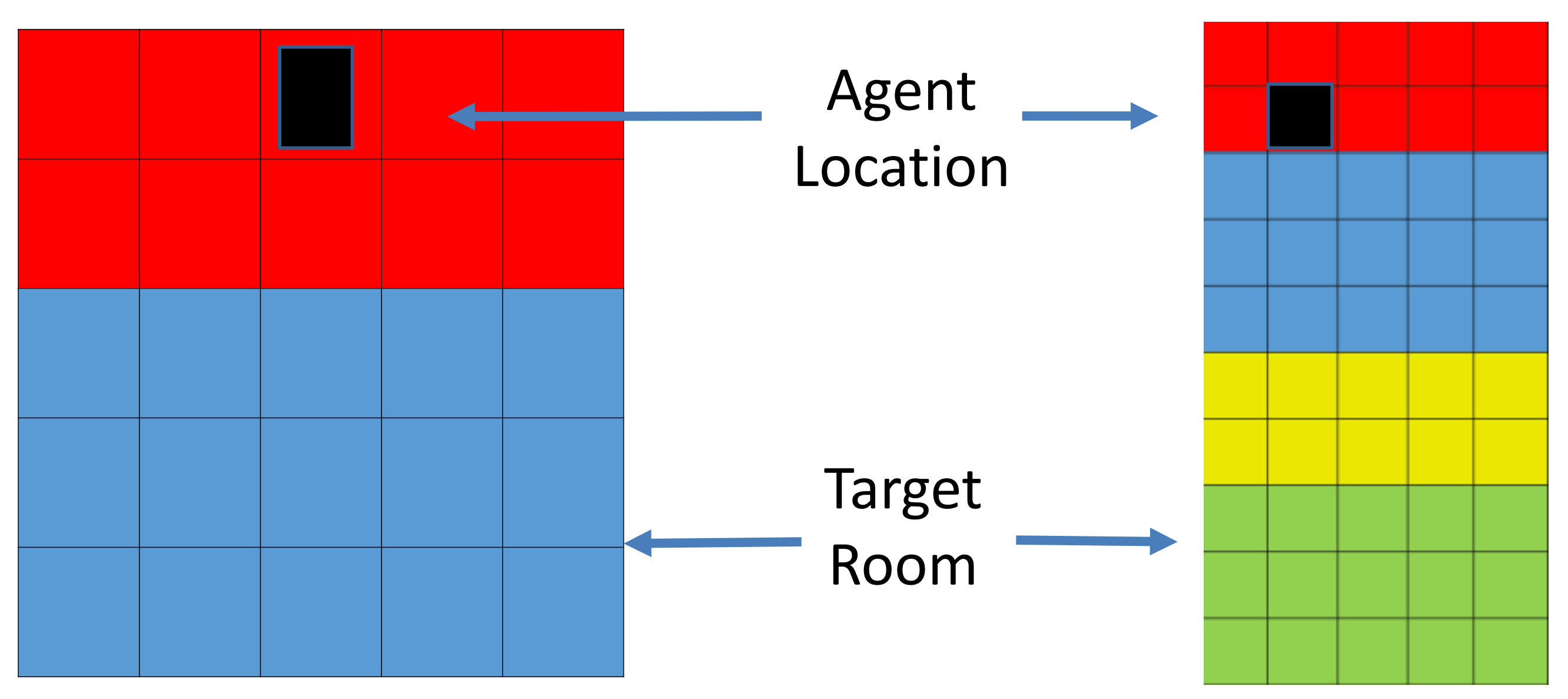


Figure 1. Domain used for meta-controller experiments.

Objectives

1) Transfer knowledge an agent has learned from one (simple) task to another (complex task) to speed up the learning process

2) Decompose the state space into sub-regions the agent knows how to solve.

Approach

Attention Mechanism

We design an attention mechanism that, at any given time step, looks at a 5x5 region of the state space. A meta-controller must learn to move the attention mechanism such that it guides the agent to where it is supposed to go.

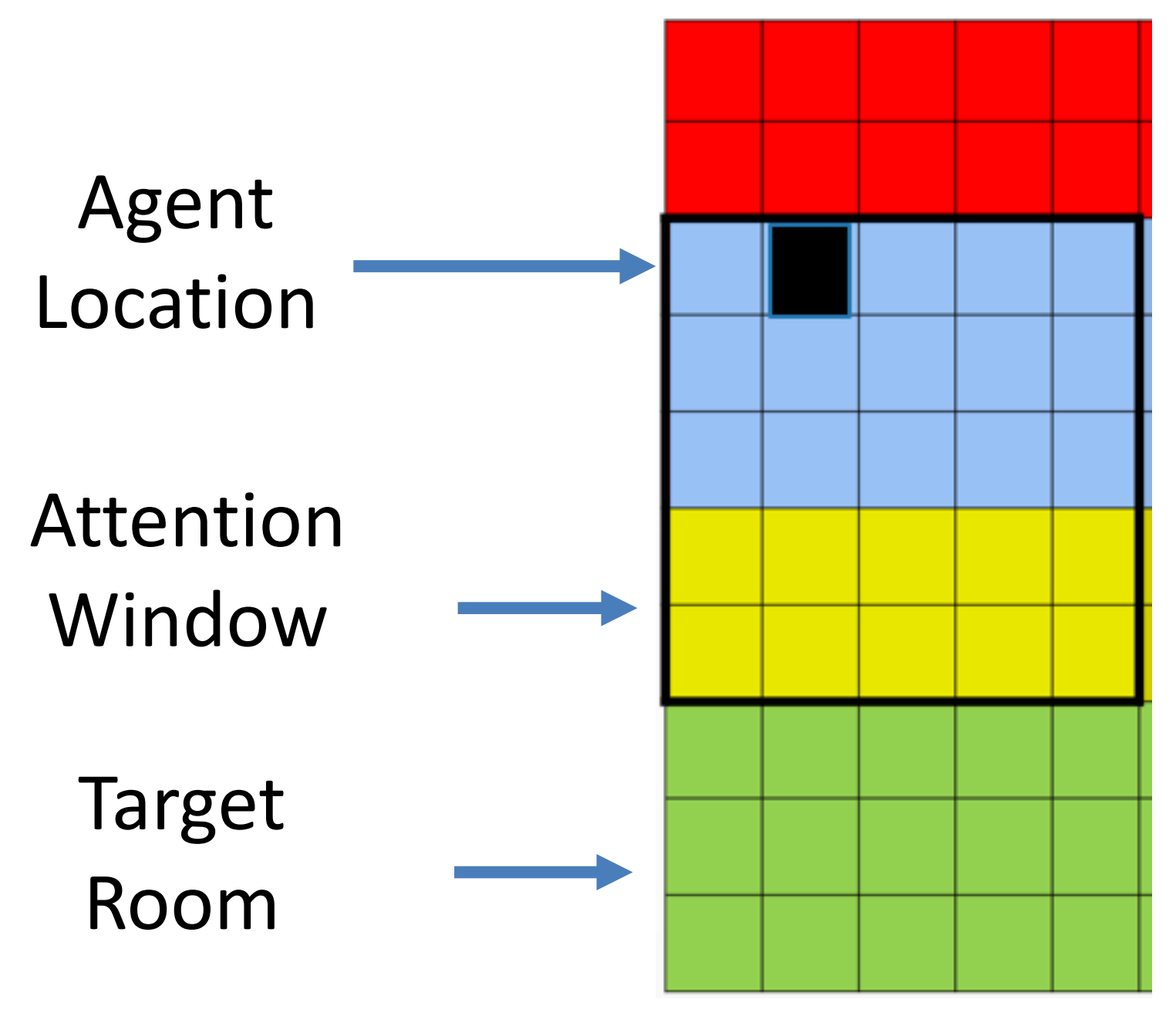
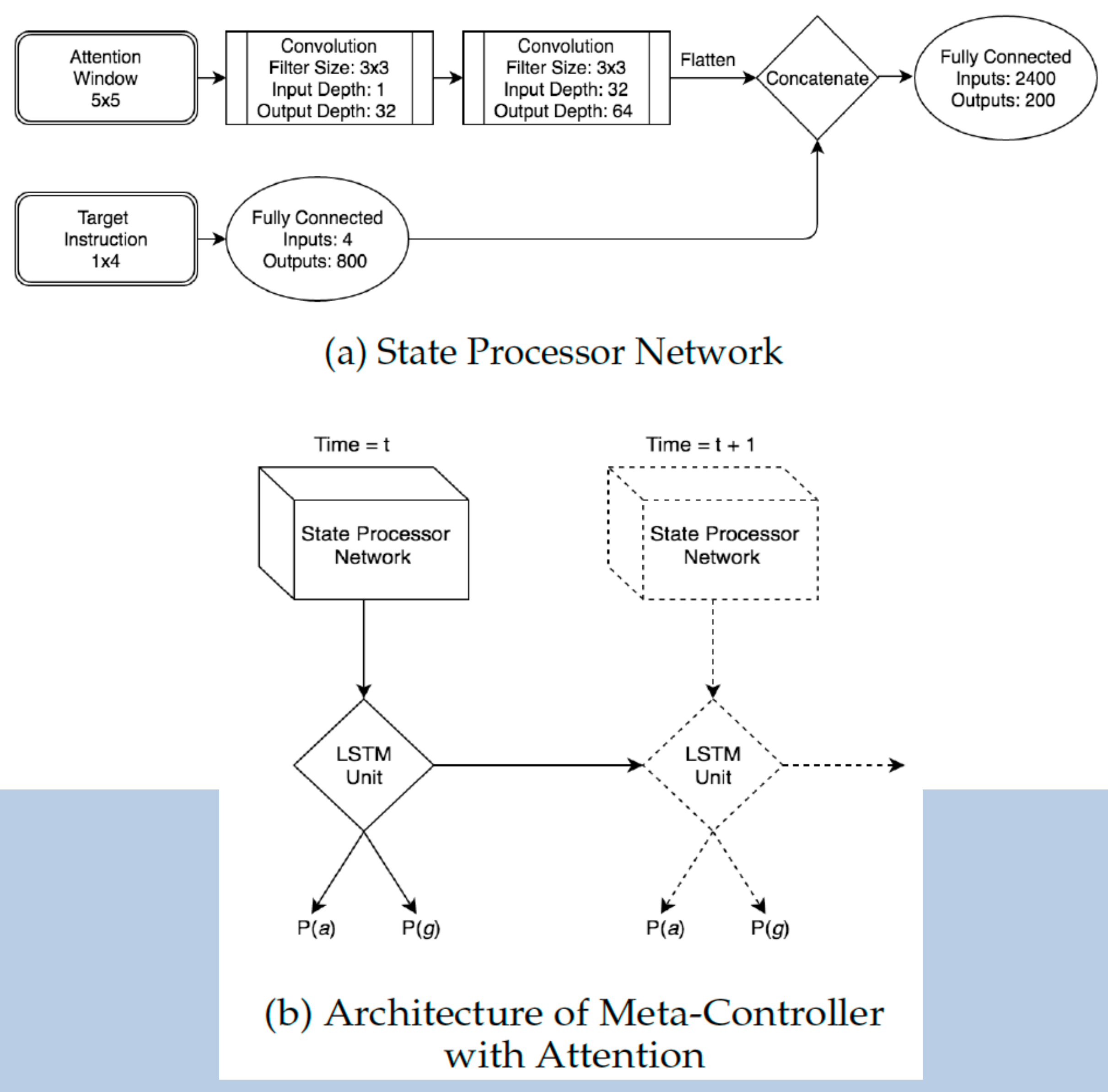


Figure 2. Illustration of Attention Mechanism.

Designing the Meta-Controller

At each time step, our meta-controller takes as input the attention window, sentence instruction, and hidden state from the previous time step. It outputs a distribution over attention actions, a distribution over sub-goals, and the next hidden state.


Attention Window 5x5
Convolution Filter Size: 3x3 Input Depth: 1 Output Depth: 32
Convolution Filter Size: 3x3 Input Depth: 32 Output Depth: 64
Flatten
Concatenate
Fully Connected Inputs: 2400 Outputs: 200
Target Instruction 1x4
Fully Connected Inputs: 4 Outputs: 800
Time = t
Time = t + 1
State Processor Network
State Processor Network
LSTM Unit
LSTM Unit
P(a)
P(g)
P(a)
P(g)

(a) State Processor Network

(b) Architecture of Meta-Controller with Attention

Figure 3. Meta-controller framework

Results

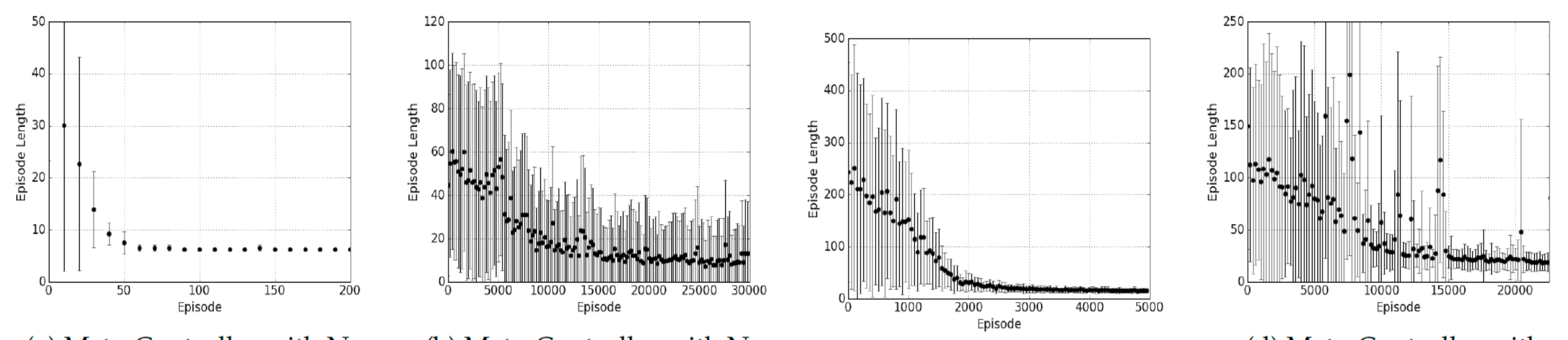
Highlighted Experiments

Meta-Controller with no Attention – Baseline experiment in which no attention mechanism is used

Constrained Attention Mechanism – Agent must be within the attention mechanism in order for the meta-controller to give it an instruction

Attention Mechanism Desired Behavior

1. Move attention to locate target room
2. Move attention back to agent's location
3. Tell agent where to go inside of the attention: wait for agent to move
4. Once agent goes to that location, moves attention in agent's direction
5. Repeat steps 3 and 4 until agent reaches target room



(a) Meta-Controller with No Attention Mechanism on a Fixed Environment

(b) Meta-Controller with No Attention Mechanism on a Dynamic Environment

(c) Meta-Controller with Partial State Decomposition

(d) Meta-Controller with Constrained Attention Mechanism

Figure 4. Episode lengths over training episodes. The dots represent means over a variable number of episodes dependent on the total number of episodes displayed and the whiskers represent the variance. In the case of the meta-controller with no attention (a and b), it converges quickly on a fixed environment and takes longer to converge in the dynamic case. The meta-controller with attention plots (c and d) show the results on the effect of using attention to guide sub-goal creation. The environment was kept fixed for these experiments.

Conclusions

Our overall contribution is a framework that allows an agent to complete a task in a large environment given knowledge of how to do so in a smaller environment. Through the use of an attention mechanism, we achieve the following:

- Smaller networks are required, which are easier to train.
- Meta-controller learns the representation of the room colors and how that representation transfers to sub-instructions that lead the agent to the desired goal.
- Scale policy learned on a smaller environment by decomposing a large state space using attention.

Contact

Saurabh Kumar	skumar311@gatech.edu
Himanshu Sahni	himanshu@gatech.edu
Farhan Tejani	farhantejani@gatech.edu
Yannick Schroecker	yannickschroecker@gatech.edu
Charles Isbell	isbell@cc.gatech.edu

References

1. Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. Vol. 1. No. 1. Cambridge: MIT press, 1998.
2. Sutton, Richard S., et al. "Policy gradient methods for reinforcement learning with function approximation." *NIPS*. Vol. 99. 1999.
3. Kulkarni, T. D., Narasimhan, K. R., Saeedi, A., & Tenenbaum, J. B. (2016). Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. *arXiv preprint.*
4. Mnih, Volodymyr, Hees, Nicolas, Graves, Alex, and Kavukcuoglu, Koray. Recurrent models of visual attention. In *NIPS,* 2014.